ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA DOPRAVNÍ

Kryštof Richter

VYUŽITÍ FORDOVA ALGORITMU PRO PLÁNOVÁNÍ LETOVÝCH TRAS

Bakalářská práce

2023

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K617..... Ústav logistiky a managementu dopravy

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Kryštof Richter

Studijní program (obor/specializace) studenta:

bakalářský – LOG – Logistika a řízení dopravních procesů

Název tématu (česky): **Využití Fordova algoritmu pro plánování letových tras**

Název tématu (anglicky): Application of the Ford Algorithm in the Issue of Flight Planning

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte následujícími pokyny:

- Představení problému, jeho aktuálnost a motivace k řešení
- Fordův algoritmus a jeho modifikace
- Analýza současného systému pro plánování letových tras
- Aplikace Fordova algoritmu na řešený problém s využitím softwaru Matlab
- Zhodnocení dosažených výsledků a aplikační metody

ČVUT FD

Rozsah grafických prací: podle pokynů vedoucích bakalářské práce

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: DURAND, Nicolas a kolektiv. Metaheuristics for Air Traffic Management. Wiley, 2016. ISBN 978-1-84821-810-9.

BANG-JENSEN, Jorgen; GUTIN, Gregory. Digraphs: Theory, Algorithms and Applications. Springer-Verlag, 2000. ISBN 978-1-84800-997-4.

Vedoucí bakalářské práce: **doc. Ing. Denisa Mocková, Ph.D.**
Ing. Karel Ječmen

Datum zadání bakalářské práce: **30. září 2022**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **7. srpna 2023**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE Fakulta dopravní
L. S.

doc. Ing. Tomáš Horák, Ph.D.
vedoucí
Ústavu logistiky a managementu dopravy

prof. Ing. Ondřej Přibyl, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

Kryštof Richter
jméno a podpis studenta

V Praze dne ..... 30. září 2022

Poděkování

Na tomto místě bych chtěl poděkovat všem, kteří mi při studiu a tvorbě bakalářské práce pomáhali. Zvláště pak děkuji Ing. Karlu Ječmenovi za odborné vedení a konzultování práce a poskytnutí přístupu k důležitým informacím a doc. Ing. Denise Mockové Ph.D. za odborné vedení a konzultování praktické části bakalářské práce.

Prohlášení

Nemám závažný důvod proti užívání tohoto školního díla ve smyslu § 60 zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některýchzákonů (autorský zákon).

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 17. června 2023 Kryštof Richter _____

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

Fakulta dopravní

Využití Fordova algoritmu pro plánování letových tras

Bakalářská práce

červen 2023

Kryštof Richter

Abstrakt

Předmětem bakalářské práce „Využití Fordova algoritmu pro plánování letových tras“ je analýza možnosti využití metody Bellman-Fordova algoritmu pro hledání minimální cesty v grafu na problematiku plánování letových tras či optimalizace tras existujících, a to zejména v rámci koncepce Free Route Airspace, jelikož dle organizace Eurocontrol bude právě toto prostředí od počátku nadcházejících let výchozím stavem ve vzdušném prostoru nad Evropskou Unií. Cílem práce je vytvoření vhodných modifikací Fordova algoritmu a jeho vstupních parametrů pro možné praktické aplikace pro plánování letových tras, naprogramování funkčního kódu zohledňující tyto modifikace a jeho následné využití na výpočet demonstrativního příkladu.

Abstract

The subject of the bachelor's thesis "Utilization of Ford's Algorithm for Flight Route Planning" is the analysis of the possibility of using the Bellman-Ford algorithm for finding the shortest path in a graph for flight route planning or optimization of existing routes, particularly within the concept of Free Route Airspace which, According to the Eurocontrol organization, will be the default state of the airspace above the European Union in the near future. The aim of the thesis is to create suitable modifications of the Ford's algorithm and its input parameters for potential practical applications in flight route planning, to program functional code incorporating these modifications, and to subsequently utilize it to calculate a demonstrative example.

Klíčová slova

Letecká doprava, doprava, optimalizace, Fordův algoritmus, Bellman-Fordův algoritmus, modifikace algoritmu, meteorologické jevy, plánování letových tras, vzdušný prostor volných tratí, programování

Key words

Air transport, transport, optimization, Ford algorithm, Bellman-Ford algorithm, Algorithm modification, Meteorological phenomena, Flight route planning, Free Route Airspace, programming

Obsah

Seznam Zkratek

Single European Sky	SES
Functional Airspace Block	FAB
International Air Transport Association	IATA
International Civil Aviation Organization	ICAO
Evropská Únie	EU
Meteorological Aerodrome report	METAR
Česká republika	ČR

Úvod

Letecká doprava je jedním ze zásadních pilířů moderního světového obchodu a cestování – a to jak pro osobní, tak pro nákladní dopravu. Rychlost, pohodlí a dostupnost, které letectví poskytuje, jsou nezbytné pro propojení odlehlých destinací navzájem či s důležitými centry a podporu mezinárodního obchodu a turismu. Nicméně, efektivní provoz letového provozu vyžaduje nejen dokonalou koordinaci mezi leteckými společnostmi, letišti a řídícími věžemi, ale také optimalizaci letových tras.

Plánování letových tras je jedním z klíčových aspektů letecké dopravy umožňující tvorbu nezanedbatelných úspor například pro parametry fyzická délka trasy, doba trvání letu či spotřeba paliva, které všechny vedou k úsporám provozních nákladů a umožňují například obsluhu dříve nevhodných lokací, což zejména pro letecké společnosti vede k vyšší konkurenceschopnosti na trhu. Optimalizace plánování letových tras je tedy zásadní pro minimalizaci nákladů, zkrácení doby letu, nebo zvýšení kapacity letecké dopravy. Důsledně plánované trasy přispívají k efektivitě leteckého provozu, minimalizaci zpoždění a zlepšení spokojenosti cestujících. Optimalizace letových tras je také klíčovým faktorem pro řízení logistiky a přepravy nákladů, přičemž přispívá k urychlení přepravy zboží a zajištění bezproblémového průběhu dodavatelského řetězce.

Inovace v oblasti plánování letových tras jsou zejména v prostorách Evropské Unie definovány novodobým zaváděním Free Route Airspace, jehož hlavním cílem je taktéž maximalizace efektivity letových tras, využití kapacity letadel a minimalizace dopadů na životní prostředí (zejména v podobě emisí). Je tedy kladen maximální důraz na využití efektivních způsobů právě plánování letových tras v rámci tohoto nově definovaného prostředí.

Plánování letových tras je složitý problém, který vyžaduje analýzu různých faktorů, jako jsou geografické podmínky, provozní omezení, meteorologické podmínky, dostupnost letišť a potřeby cestujících. Řešením úloh aplikovatelných na tuto problematiku se zabývá jedna z disciplín Operačního výzkumu – Teorie grafů, která poskytuje vhodný matematický rámec pro modelování leteckých sítí a hledání optimálních tras. Jedním ze základních problémů, které Teorie grafů řeší, je nalezení minimální cesty grafem, jejíž metody lze právě do oblasti plánování letových tras snadno adaptovat.

Existuje několik algoritmů pro hledání minimálních cest v grafu, jedním z nichž je Fordův algoritmus. Fordův algoritmus, pojmenovaný po matematikovi L.R. Fordovi, je efektivní algoritmus pro nalezení minimálních cest či drah mezi dvěma vrcholy v orientovaném nebo neorientovaném grafu. Tento algoritmus je založen na principu postupného relaxování hran

grafu a aktualizování délky cesty ke každému vrcholu. Fordův algoritmus je dobře známý pro svou schopnost pracovat s ohodnocenými hranami, což je přesně to, co je potřeba pro plánování letových tras.

Cílem této bakalářské práce je zkoumat využití Fordova algoritmu jako nástroje pro plánování letových tras a představit Fordův algoritmus jako nástroj pro plánování letových tras a poskytnout jeho hodnocení z hlediska efektivity, výhod a omezení v reálném světě letecké dopravy. Dále budou identifikována případná vylepšení a rozšíření tohoto algoritmu pro další optimalizaci provozu letových tras a podporu rozvoje letecké dopravy do budoucnosti.

1 Letecká doprava

Letecký prostor je definován jako prostor nad zemským či mořským povrchem převážně určen k provozu letecké dopravy, a to Nařízením o společných pravidlech pro provozování letecké dopravy v EU (nařízení EU č. 1008/2008), ve kterém je zároveň vymezena řada základních pravidel pro provoz letecké dopravy v rámci EU [22].

Na základě nařízení o Jednotném Evropském nebi (Single European Sky) - SES nařízení EU č. 549/2004 - byl stanoven rámec pro řízení a využívání letového prostoru Evropské Unie s hlavním cílem zvýšení efektivity letecké dopravy a došlo k vytvoření oblasti pro jednotné řízení letového provozu (nazvané právě SES), usilující o harmonizaci a optimalizaci letového provozu v prostorách EU [23].

Na základě těchto nařízení vyplývá pro provozování letecké dopravy dosažení několika cílů:

- dosažení požadovaného výkonu,
- dodržení bezpečnostních požadavků,
- minimalizace dopadů na životní prostředí.

Jedním ze způsobů, jak těchto cílů dosáhnout, je optimalizace klíčových parametrů (jako jsou např. délka cesty (vzdálenostní či časová)) při plánování letových tras.

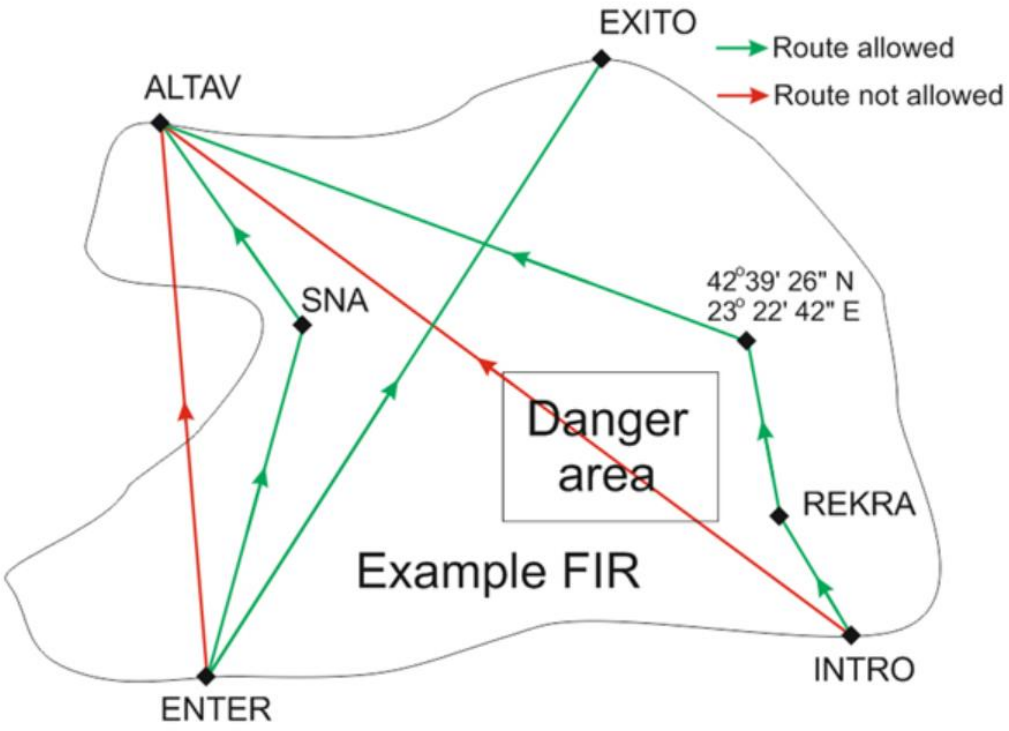
2 Free Route Airspace (FRA)

Ve snaze právě umožnit či usnadnit optimalizaci plánování letových tras vznikl v rámci EU koncept Volného vzdušného prostoru – Free Route Airspace (FRA), což je vytyčený vzdušný prostor, v jehož mezích mohou uživatelé volně plánovat svou trasu mezi definovaným vstupním a výstupním bodem. Na základě dostupnosti a obsazenosti prostoru lze trasu plánovat přímo nebo za využití oficiálních či neoficiálních orientačních bodů. Tímto přestává být nutné odkazovat se na existující síť letových tras. [15]

Cílem konceptu je snaha snížit množství omezení, kterým uživatel plánující pohyb prostorem podléhá. Plánování letových tras v těchto oblastech přesto podléhá určitým pravidlům, mezi které patří zejména:

- využití daných vstupních a výstupních bodů,
- vyhnutí se zakázaným, omezeným a nebezpečným oblastem,
- nutnost celé trasy od vstupního bodu po výstupní bod zůstat v mezích FRA.

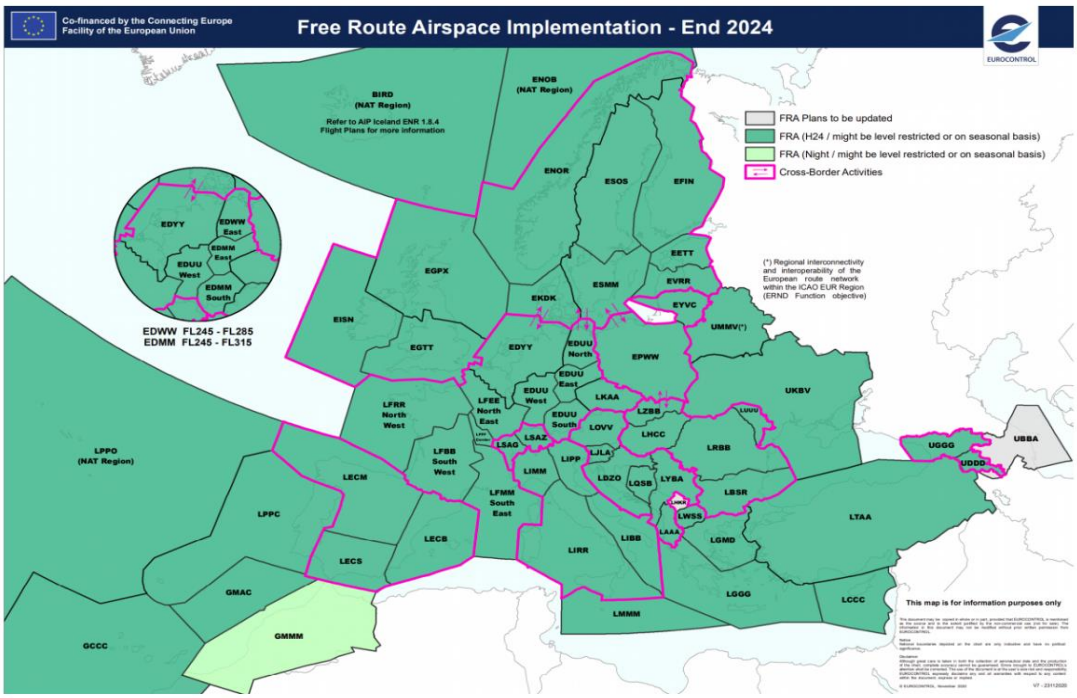
Grafické znázornění těchto pravidel se nachází na obrázku [1].



Obrázek 1 – Schéma pravidel a omezení využití FRA [zdroj: 15]

Hlavní výhody využití FRA spočívají ve významně vyšší možnosti optimalizovat letové trasy, tudíž také dobu letu a spotřebu paliva, čehož je dosaženo umožněním přímých letů mezi destinacemi.

V současné době je cílem organizace Eurocontrol zavést prostory FRA ve vzdušném prostoru nad většinou Evropy (podrobný plán znázorňuje Obrázek 2), pro problematiku letového plánování (alespoň tedy pro oblast Evropského kontinentu) je vhodné brát tuto skutečnost v potaz a vytvářet řešení aplikovatelná v rámci FRA. [13]



Obrázek 2 – Mapa plánovaného stavu FRA v rámci EU podle organizace Eurocontrol [zdroj: 15]

Klasifikace FRA

V současné době je definováno několik typů FRA [15]:

- časově omezené – pro tento typ FRA platí, že daná oblast slouží jako FRA pouze ve specifikovaných časových úsecích,
- geograficky omezené – pro tento typ FRA platí, že existují horizontální či vertikální hranice, mimo které se letadla nemohou pohybovat,
- v rámci Functional Airspace Block (FAB),
- v rámci Single European Sky (SES),

Klíčové body FRA

V rámci FRA dále existuje několik klíčových bodů pro správné užití a orientaci v prostoru:

- Horizontal Entry Point (E) – bod nacházející se na hranici daného FRA (mimo mimořádné situace, kdy neexistuje praktické alternativní řešení ohledně lokace, umožňující tento bod umístit uvnitř či vně hranice daného FRA), jehož prostřednictvím je umožňován vstup do daného FRA [13],

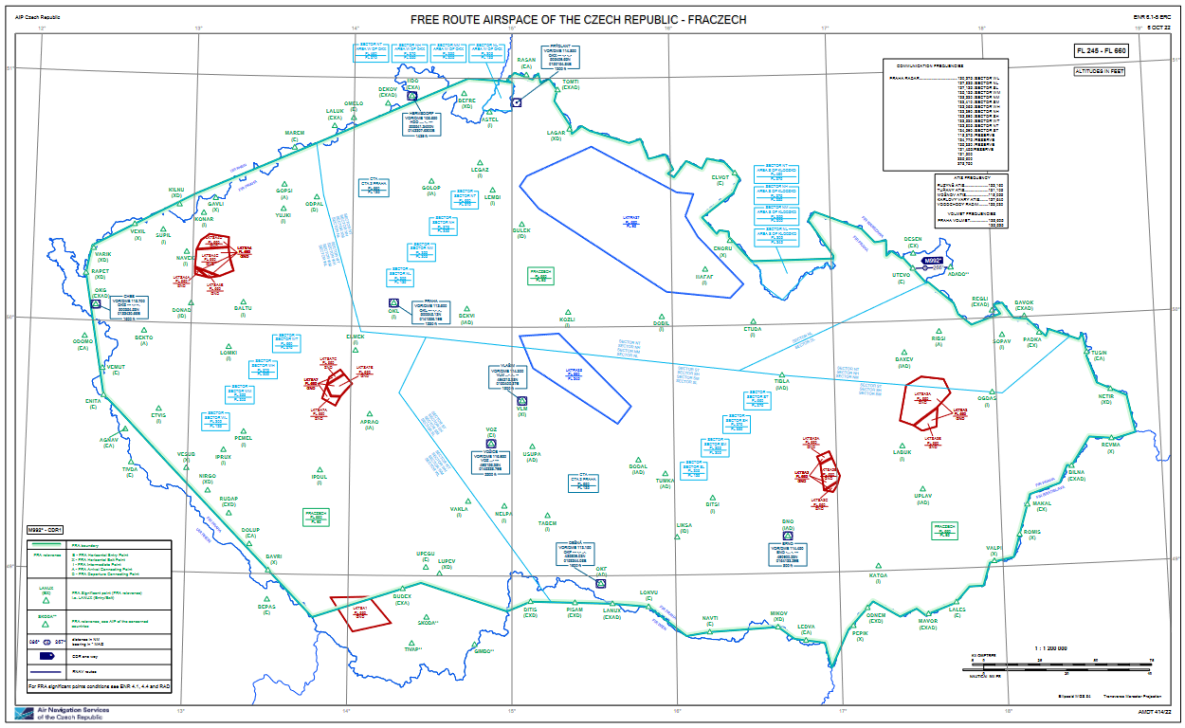
- Horizontal Exit Point (X) – bod nacházející se na hranici daného FRA (mimo mimořádné situace, kdy neexistuje praktické alternativní řešení ohledně lokace, umožňující tento bod umístit uvnitř či vně hranice daného FRA), jehož prostřednictvím je umožňován výstup do daného FRA [13],

- Intermediate Point (I) – bod nacházející se uvnitř hranic daného FRA (resp. ležící mezi body E a X), přes nějž je možno plánovat letové trasy v rámci FRA; body mohou být publikované, nebo nepublikované [13],

- Arrival Connecting Point (A) – bod definovaný v závislosti na letiště nacházející se buď uvnitř existujícího FRA prostoru, v některých případech pak nacházející se v hraniční oblasti FRA či zcela mimo FRA, který umožňuje operace FRA pro přistávající leteckou dopravu daného letiště; pro vertikální rozdělení prostoru lze Arrival Connecting Point (A) považovat zároveň také jako Horizontal Exit Point (X)[13],

- Departure Connecting point (D) – bod definovaný v závislosti na letiště nacházející se buď uvnitř existujícího FRA prostoru, v některých případech pak nacházející se v hraniční oblasti FRA či zcela mimo FRA, který umožňuje operace FRA pro odlétající leteckou dopravu daného letiště; pro vertikální rozdělení prostoru lze Departure Connecting Point (D) považovat zároveň také jako Horizontal Entry Point (E) [13].

Demonstrativní uspořádání prostoru FRA znázorňuje Obrázek 3, kde lze pozorovat rozmístění jednotlivých klíčových bodů FRA a hranice prostoru volně kopírující skutečné státní hranice českého státu.



Obrázek 3 – Mapa FRA České republiky pro letové hladiny FL245 – FL660 [zdroj: 19]

3 Organizace IATA

Jednou z organizací figurujících v provozu letecké dopravy je Mezinárodní asociace leteckých dopravců, která sdružuje letecké dopravce (na 280 společností z celého světa). Mezi její povinnosti spadá například stanovování podmínek pro přepravu nebezpečných nákladů, dohled nad vzájemným účtováním mezi jednotlivými členy a přidělování kódů IATA jednotlivým letištím.

Kód IATA

Kód IATA je třímístný kód, který na základě parametrů stejnojmenné organizace definuje geografické lokace – nejčastěji letiště. Jednou z významných aplikací kódu je identifikátor na visačkách pro zavazadla, který slouží pro identifikaci destinace daného zavazadla a je tudíž používán jak pro základní manipulaci, tak jako nástroj pro vracení ztracených kufrů.

Kódy IATA nemají pevnou strukturu, existuje však několik obecných způsobů, na jejichž základě jsou kódy tvořeny:

- první tři písmena města, ve kterém se letiště nachází (např. IST pro Istanbul),
- kombinace libovolných tří písmen sídelního města (např. WAW pro Varšavu),

- kód Národních meteorologických služeb (NWS) pro některá města v USA doplněná o písmeno X (např. LAX pro Los Angeles),
- kombinace písmen ze samotného názvu letiště v případech, kdy letiště obsluhuje více měst, případně pokud se ve městě nachází více letišť (např. JFK pro letiště Johna F. Kennedyho v New Yorku nebo CDG pro letiště Charlese de Gaulla v Paříži),
- kombinace písmen založená na názvech několika měst/regionů, které dané letiště obsluhuje (např. DFW pro Dallas/Fort Worth). Letištím, která obsluhují více zemí, lze být přiřazeno více kódů (např. Letiště Basel-Mulhouse-Freiburg s kódy BSL, MLH a EAP).

K zachování kódů dochází často i při změně názvu města či letiště – toto vysvětluje případy, na které nelze aplikovat předchozí konvence.

4 Organizace ICAO

International Civil Aviation Organization nebo také Mezinárodní organizace pro civilní letectví je nadnárodní organizace, která sdružuje principy, strategie a techniky mezinárodní letecké navigace a podporuje rozvoj mezinárodní letecké dopravy s důrazem na bezpečný růst.

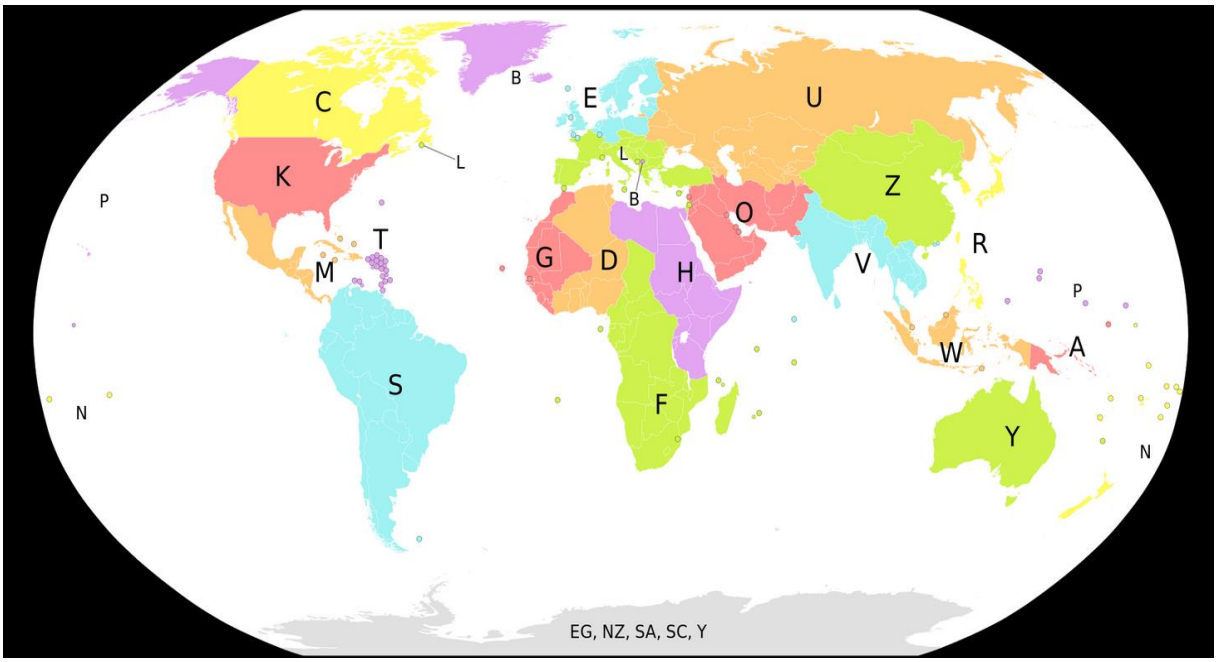
Organizace nastavuje standardy a doporučené postupy a chování v oblastech navigace, letecké infrastruktury, letové inspekce, prevence nezákonného zasahování do leteckého provozu, zavádí procedury pro mezinárodní pohyb a přesun přes státní hranice a definuje protokoly pro vyšetřování leteckých nehod, které jsou dodržovány zmocněnými úřady ve všech signatářských zemích Chicagské konference. Samotné kódy jsou zpravidla využívané zejména Řízením letového provozu a v procesech typu letového plánování. Pro potřeby bakalářské práce spočívá jejich výhoda oproti kódům IATA zejména ve dvou aspektech:

- zprávy METAR, které jsou v bakalářské práci aplikované jsou vydávány pouze pod kódy ICAO,
- kódy ICAO bývají přiřazovány také strukturám (jako jsou například meteorologické stanice) nacházejícím se mimo pozemky letiště, což umožňuje větší svobodu v konstrukci sítě či grafu pro tvorbu letového plánu [14].

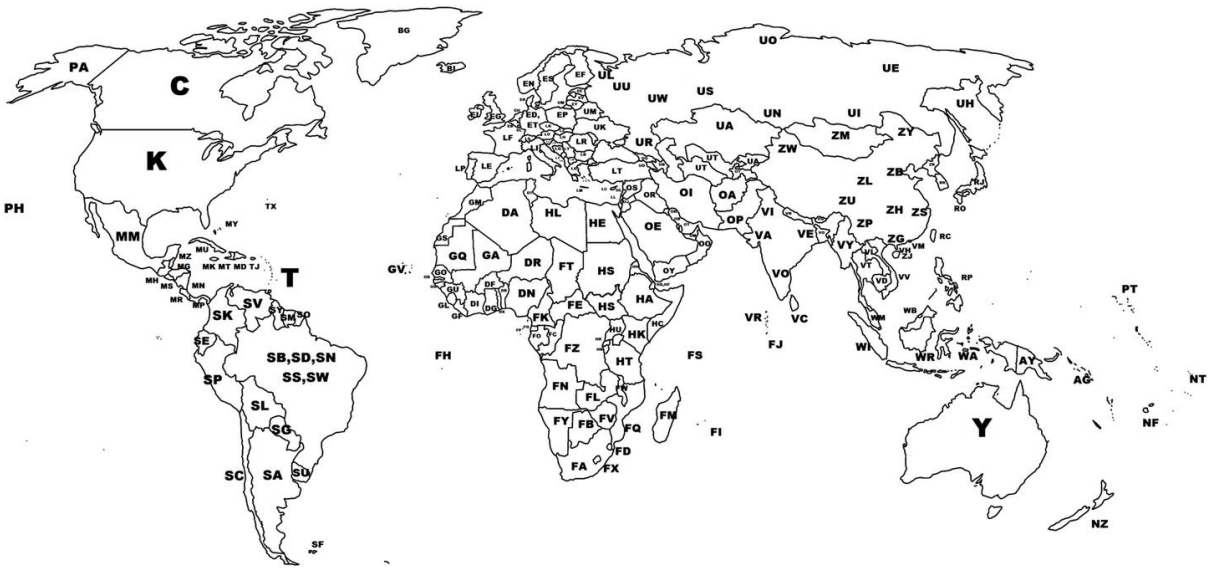
Kód ICAO

Kódy ICAO jsou na rozdíl od kódů IATA čtyřmístné a pevně strukturované. Obvykle korespondují první dva znaky k písmenům identifikujícím region (první znak, mapa písmen znázorňuje Obrázek 4) a stát, případně podoblast v daném státě (druhý znak, mapa písmen

znázorňuje Obrázek 5), přičemž následující dva znaky identifikují samotné letiště. Například Letiště Václava Havla v Praze spadá pod ICAO kód LKPR.



Obrázek 4 – Mapa korespondujících písmen pro první znak kódu ICAO [zdroj: 16]



Obrázek 5 – Mapa korespondujících písmen pro druhý znak kódu ICAO [zdroj: 16]

Jelikož dvoumístná sekvence koncové části kódů umožňuje existenci pouze velmi omezeného počtu kombinací, dochází v některých případech (zejména s vysokým počtem letišť) ke tvorbě šestimístných pseudo-kódů, kde první dva znaky sdílejí strukturu s klasickým ICAO kódem, následující dva znaky určují geografickou správní jednotku v daném státě

(například „oddělení“ ve Francii, kde jsou pseudo-kódy často použity, jelikož se v zemi nachází 629 letišť) a poslední dva znaky udávají číselné pořadí struktury.

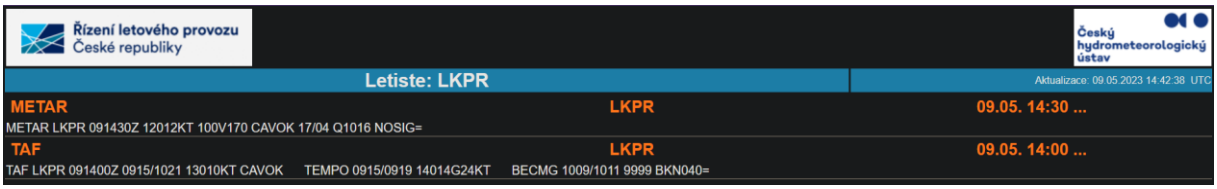
5 METAR

Meteorological Terminal Air Report je formát zprávy, který se využívá pro pravidelné hlášení ohledně současného stavu počasí, a to zejména piloty a složkami řízení letového provozu, ale také například meteorology při předpovídání počasí.

Informace pro METAR zprávy jsou zpravidla získávány na permanentních meteorologických stanovištích umístěných především na letištích nebo ve vybraných klíčových bodech.

Obsah zprávy je generován pravidelně každou hodinu, na rušných letištích pak každých třicet minut a ve zvláštních případech každých dvacet minut, může být vytvořen kompletně automaticky strojově, nebo naměřen přístroji a následně sestaven meteorologem.

V případě náhlé změny počasí v mezidobí pravidelné generace zpráv bývá vydávána speciální zpráva typu SPECI, která má ovšem stejný formát.



Obrázek 6 – Příklad vydávané zprávy METAR organizací ŘLP [zdroj: 17]

Formát zprávy METAR

Pro demonstraci byla použita náhodně vybraná zpráva METAR.

METAR EDDL 161450Z AUTO VRB01KT 1000 R23R/1500D R23L/1400N BCFG OVC002 M01/M02 Q1017 TEMPO 0600 FZFG

Struktura zprávy (jejíž příklad znázorňuje Obrázek 6) vypadá obvykle následovně:

METAR lokace čas vítr dohlednost jevy oblačnost teplota tlak dráhy předpověď poznámky =

Jednotlivé části zprávy METAR tedy znamenají:

- METAR – typ zprávy (alternativou je označení SPECI),
- EDDL – čtyřmístný kód ICAO daného letiště (v tomto případě se jedná o Mezinárodní letiště Düsseldorf),

- 161450Z – datum a čas měření, kde první dvě číslice značí den v měsíci, následující čtyři číslice značí přesný čas měření a poslední znak stanovuje, že čas měření je udáván v koordinovaném světovém čase UTC (označovaném také jako „Zulu time“, v zprávě je tedy popsán písmenem Z) – v tomto případě tedy byla zpráva vytvořena

- AUTO – způsob vygenerování zprávy – v tomto případě byla zpráva vytvořena zcela automaticky,

- VRB01KT – směr a rychlost větru, kde první tři znaky udávají zeměpisný směr větru ve stupních (v případě zcela proměnlivého směru větru jsou číslice nahrazovány údajem „VRB“ neboli „variable“, v překladu „proměnlivé“), následující dvě číslice udávají rychlost větru a poslední dva znaky určují jednotky (obvykle jsou využívány uzly (KT), alternativně pak metry za sekundu (MPS) či kilometry za hodinu (KMH)). V tomto případě je tedy směr vanutí větru zcela proměnlivý a dosahuje rychlosti 1 uzlu. V případě nižší kolísavosti směru větru bývá tato část zprávy následována doplňující sekcí ve formátu „XXXVYYY“, První tři číslice „XXX“ a poslední tři číslice „YYY“ udávají mezní hodnoty směru větru, zatímco písmeno V slouží jako dělící znak; hodnota směru větru v předchozí části zprávy je pak váženým průměrem těchto hodnot,

- 1000 – horizontální dohlednost v metrech, v tomto případě tedy dosahuje hodnota dohlednosti 1000 m. Údaj může nabývat hodnot až do výše 9999, což značí dohlednost nad 10 km; v případech, kdy je dohlednost variabilní v jednotlivých směrech bývá údaj o převládající dohlednosti doplněn o sekci ve formátu „XXXXYY“, kde číslice „XXXX“ opět značí hodnotu dohlednosti v metrech a znaky „YY“ udávají korespondující světovou stranu (například doplňující zpráva 4600NW by značila dohlednost 4600 m severozápadním směrem),

- R23R/1500D – dráhová dohlednost v metrech. Údaj před dělícím znakem „/“ značí vzletovou či přistávací dráhu, na kterou se dohlednost vztahuje, kde znak R slouží jako zkratka pro „Ranvej/Runway“, následující skupina číslic popisují konkrétní dráhu a doplňující znak „R“ (alternativně „L“) umožňuje rozlišování paralelních drah, a údaj za dělícím znakem „/“ udává hodnotu dohlednosti na dané ranveji a zároveň posledním znakem „D“ (alternativně „U“) určuje sestupnou (alternativně vzestupnou) tendenci; v tomto případě je tedy dráhová dohlednost na paralelní ranveji 23R se sestupnou tendencí rovna 1500 metrů,

- BCFG – kódy významných meteorologických jevů,

- OVC002 – oblačnost a hodnota její spodní hranice ve stovkách stop (ft), v tomto případě je počínaje výškou 200 ft zataženo (pokrytí oblohy rovno 8/8, resp. 1 celku),
- M01/M02 – hodnota současné teploty a rosného bodu; znak „M“ indikuje záporné hodnoty; v tomto případě dosahuje hodnota teploty na -1 °C a rosného bodu na -2 °C,
- Q1017 – hodnota současného tlaku vzduchu přepočítaného na hladinu moře v hektopascalech (hPa), v některých oblastech dochází namísto hPa k využití tzv. Palců rtuti (inHg), přičemž je znak „Q“ nahrazen znakem „A“; v tomto případě je tedy hodnota tlaku vzduchu rovna 1017 hPa,
- TEMPO – klíčové uvozovací slovo pro krátkodobou předpověď počasí – k následujícím údajům jsou z meteorologického hlediska očekávány přechodné a nepravidelné změny a zároveň svým formátem odpovídají hlavní zprávě; hodnoty 0600 a FZFG následující tuto část zprávy tedy značí horizontální dohlednost a významné meteorologické jevy (Podchlazení, mlha), u kterých se očekává nadcházející změna.

Klíčovými prvky pro potřeby bakalářské práce jsou kód ICAO, který umožňuje přiřadit korespondující meteorologické podmínky ke geografickým souřadnicím letiště (či jiného orientačního bodu) a údaj o rychlosti a směru větru, jehož hodnota ovlivní výsledné ohodnocení hran pro výpočet Fordovým algoritmem.

Při zvolení správných vah a ohodnocení však lze podobný výpočet provést také pro ostatní prvky zprávy METAR (např. oblačnost, dohlednost či výskyt mimořádných meteorologických jevů), tyto výpočty však představují ve srovnání s povětrnostními podmínkami komplikaci, jelikož je nelze jednoznačně převést na jednotky typu délka a trvání cesty či spotřeba paliva na daném úseku.

Seznam význačných jevů pro zprávy METAR

Intenzita

- + = silná (heavy),
- <bez označení> = mírná (moderate),
- - = slabá (light).

Popis

- SH = přeháňka (shower),

- BL = zvířený (blowing),
- DR = nízko zvířený (low drifting),
- MI = přízemní (shallow),
- BC = chuchvalce (patches),
- PR = částečné (partial),
- FR = namrzající (freezing).

Srážky

- DZ = mrholení (drizzle),
- RA = déšť (rain),
- SN = sněžení (snow),
- SG = sněhová zrna (snow grains),
- PL = zmrzlý déšť (ice pellets),
- GR = kroupy (hail),
- GS = krupky (small hail).

Jevy ovlivňující dohlednost

- FG = mlha (fog),
- BR = kouřmo (mist),
- SA = písek (sand),
- DU = prach (dust),
- HZ = zákal (haze),
- FU = kouř (fume),
- VA = vulkanický popel (volcanic ash).

Ostatní jevy

- TS = bouřka (thunder storm),

- PO = prachové/písečné víry (dust/sand whirls),
- SQ = húlava (squal),
- FC = nálevkovitý oblak (funnel cloud),
- DS = prachová vichřice (dust storm),
- SS = písečná vichřice (sand storm),
- VC = v blízkosti (vicinity).

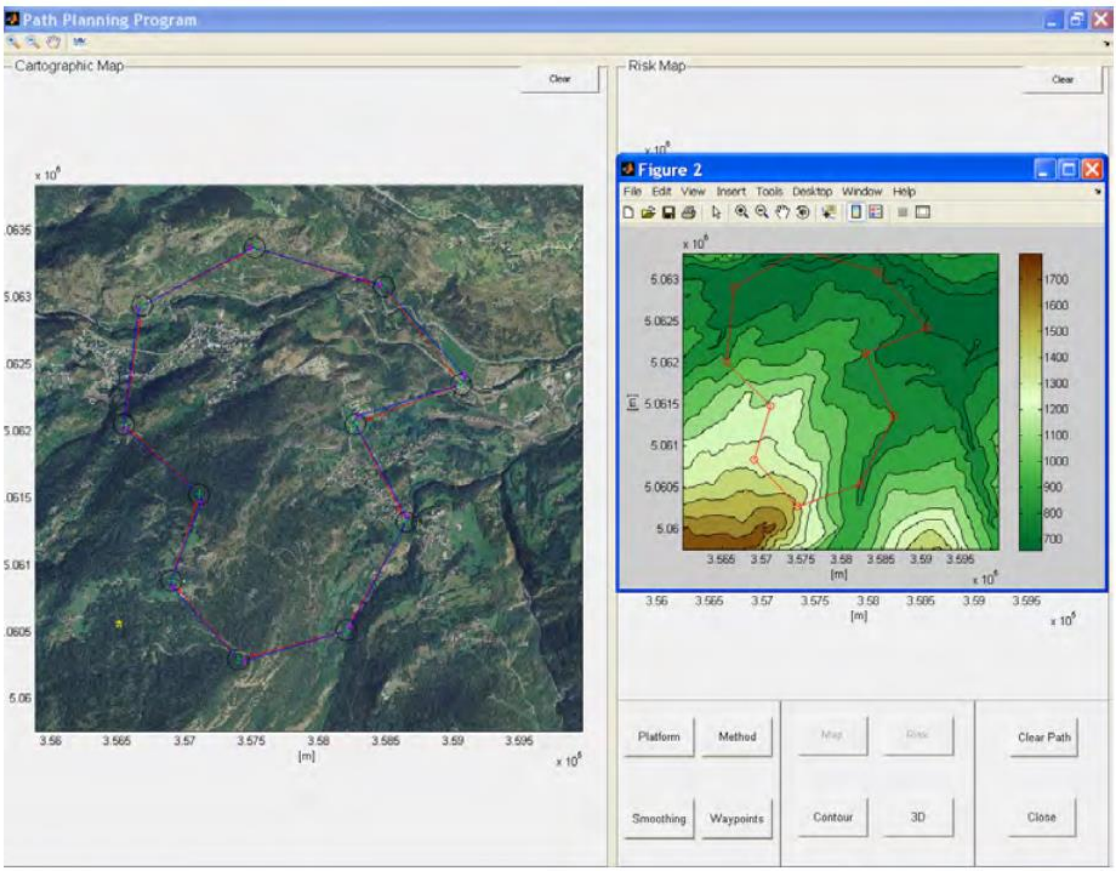
6 Stávající způsoby plánování letových tras

Cílem plánování letových tras zpravidla bývá vytvoření trajektorie ve skutečném časoprostoru mezi bodem vzletu letadla a bodem přistání letadla (případně s inkorporací mezilehlých bodů) tak, aby došlo k zabránění kolizím, vyhnutí se překážkám a byly dodrženy základní zásady letových pravidel, ale také zároveň došlo k optimalizaci funkce určitého letového parametru (například doba letu, délka trajektorie, spotřeba paliva apod.). Pro tyto účely byla vyvinuta řada metod, s různorodými konečnými cíli – minimalizace risku, vyhnutí se překážkám či maximalizace výkonu. Každá z těchto metod využívá řadu prvků z různých vědních disciplín od fyziky, přes matematiku až po počítačové vědy.

6.1 Manuální plánování pomocí softwaru PCube

První a základní způsob plánování letových tras byl z důvodu nemalé komplexity, přísných požadavků na bezpečnost a nízkých kapacit podpůrných technologií prováděn manuálně. Na základě prostředí, kterým trasa byla vedena, a potřeb navštívených lokací byly ručně vytvořeny v daném programu a následně načteny do letových kontrolních systémů letadel.

Pro tvorbu takovýchto tras a řešení s nimi spojených problémů došlo například k vytvoření balíčku PCube pro software Matlab a Simulink. Tento program byl následně využíván k tvorbě a generaci tras na základě zadaných požadavků. Mezi funkce programu patřila například generace tvarů z databáze programu či zcela manuální definice orientačních bodů. Přímou návazností na tento způsob plánování letových tras se stala tzv. Dubinsova křivka, jejíž aplikace i nadále hrají důležitou roli v optimalizaci tras. V rámci programu PCube zohledňuje aplikace Dubinsových křivek rychlost letadla a jeho schopnost manévrovat, díky čemuž je dosaženo tvorby časově a distančně kratších tras obsluhující dané sítě a grafy. Příklad této aplikace Dubinsových křivek znázorňuje Obrázek 7, kde za jejich použití dochází k optimalizaci letové trasy, která byla předem manuálně nadefinovaná uživatelem. [3], [4], [5]



Obrázek 7 – Aplikace Dubinsových křivek v programu PCube [zdroj: 1]

6.2 Pravděpodobnostní algoritmy

Pravděpodobnostní algoritmy, které jsou mimo problematiku plánování letových tras hojně využívány v nejrůznějších vědeckých disciplínách (např. Machine learning, kryptografie, analýza dat apod.), fungují na základě inkorporace určité míry náhodnosti do svého procesu za účelem dosažení vyššího výkonu. V nejširším slova smyslu se tyto algoritmy dělí na dva základní typy:

- algoritmy, jejichž vstup je náhodný, čas běhu konečný a výstup optimální,
- algoritmy, jejichž výstup optimální být nemusí a jejichž čas běhu může být nekonečný (resp. nedojde k terminaci algoritmu).

Pro řadu problémů jsou pravděpodobnostní algoritmy jediným způsobem jejich řešení.

V rámci problematiky plánování letových tras dochází často k využití map či grafů modelujících distribuci hrozeb, překážek a zakázaných oblastí v procházeném prostředí. Často využívaný prvek těchto metod je například Markovův řetězec (popřípadě Markovův proces), což je forma stochastického modelu, pro který platí, že pravděpodobnost pro každý

další stav je založena pouze na stavu současném. Na čas lze pro tyto řetězce pohlížet jako na:

- spojitý,
- diskrétní.

Hlavní nedostatek těchto algoritmů je minimální korelace mezi vygenerovanou letovou trasou a kinematikou letadla, které by tuto trasu mělo obsloužit. Zkonstruovaná trasa v mnoha případech není pro letadlo prakticky obsloužitelná, je tudíž nutno dodatečné využití určitých „uhlazovacích“ algoritmů, které trasu upraví pro skutečný provoz, což často vede k drastickému zvýšení časové složitosti řešeného problému. [6], [7]

6.3 Algoritmy pro prediktivní řízení modelů

Princip fungování algoritmů pro prediktivní řízení modelů, které byly původně vyvinuty pro potřeby v odvětví průmyslového procesního řízení a kontroly (věda zabývající se dosažením určité úrovně průmyslové výroby za splnění parametrů typu bezpečnost, efektivita a ekonomičnost, konzistence apod), spočívá v namodelování daného systému do určité míry složitosti za využití vysokého počtu sad komplexních diferenciálních rovnic a následné ovlivňování tohoto procesu modifikací proměnných, které tento model a proces definují. Mezi tyto proměnné mohou patřit například teplota, tlak nebo rychlost. Pro potřeby plánování letových tras jsou pak v praktickém užití vstupními daty daného modelu informace senzorů a snímačů letadla

Hlavní výhodou těchto algoritmů je možnost využití dat generovaných (a tudíž dostupných) pouze v průběhu samotného letu a následná adaptace existující letové trasy pro potřeby vymezené těmito daty. Problematika aplikace těchto metod spočívá ve vysoké časové náročnosti iterativního řešení složitých soustav diferenciálních rovnic potřebného pro generaci cest ve skutečném prostředí pro plánování letových tras. S rostoucí komplexností problému zároveň roste složitost procesu potřebného k nalezení skutečně optimálního řešení, pro využití na praktické problémy je tedy klíčová aplikace heuristických metod, které však v mnoha případech neexistují. [9], [12]

6.4 Algoritmy potenciálních polí

Algoritmy potenciálních polí byly původně vyvinuty pro potřeby robotiky a jí přidružených věd, přičemž později došlo k jejich adaptaci pro plánování letových tras pouhou modifikací kinematických a překážkových modelů. V robotice zpravidla dochází k aplikaci modelů potenciálních polí s elektrickými či magnetickými parametry, pro plánování letových tras jsou mnohem vhodnější modely aerodynamické. Princip těchto algoritmů je založen na tvorbě

odpudivých sil okolo vytyčených překážek a zakázaných oblastí a přitažlivých sil okolo cíle trasy. Existují dva hlavní způsoby, na jejichž základě jsou tyto aerodynamické modely generovány:

- Metoda šířící se tlakové vlny,
- Metoda aerodynamického modelování pole pohybu.

Metoda šířící se tlakové vlny uvažuje pomyslnou vlnu šířící se směrem od cílové pozice trasy až k dosažení pozice startovní, zatímco v průběhu tohoto šíření dochází v okamžiku kolize vlny a překážky (resp. jiné zakázané oblasti) k zaznamenání této překážky do mapy procházeného prostředí. Metoda aerodynamického modelování pole pohybu tvoří reprezentaci procházeného prostoru jako aerodynamické pole, překážky (resp. jiné zakázané oblasti) jsou definovány jako určité body singularity obklopené směrovými vektory odpudivých sil a cílová lokace je zastoupena jediným bodem singularity obklopeným směrovými vektory přitažlivých sil. Hledání optimální trasy je následně prováděno minimalizací gradientu potenciálních polí. [11]

Ačkoliv tyto algoritmy představují vhodné řešení pro řadu problémů plánování tras (zejména v rámci problematiky, pro kterou je klíčové vyhýbání se překážkám), jejich aplikace na komplexní problémy je značně omezené tendencí vyhledávat spíše lokální minima. [8]

8.5 Algoritmy pro nalezení minimální cesty v grafu

Na řešení problematiky plánování letových tras a jejich optimalizace lze aplikovat také řadu metod disciplíny Operačního výzkumu – Teorie grafů. Jednou z těchto metod je aplikace algoritmů pro řešení problému nalezení minimální cesty v grafu, což je jedna ze základních úloh, kterou se Teorie grafů zabývá a Fordův (nebo také Bellman-Fordův) algoritmus, který byl zvolen jako hlavní metoda pro řešení problematiky bakalářské práce spadá mezi způsoby hledání minimální cesty grafem.

9 Fordův algoritmus [28]

Fordův algoritmus (V některých případech také označován jako Bellman-Fordův algoritmus) je algoritmus využívaný v teorii grafů pro vyhledání minimálních cest z jediného zdrojového vrcholu do všech ostatních vrcholů. Pro hledání minimální cesty mezi vrcholy v grafu jsou vedle Fordova algoritmu využívány také Dijkstrův algoritmus a Floydův algoritmus. Jednotlivé algoritmy se mezi sebou liší časovou náročností, výstupní informací a použitelností v grafech obsahující negativně ohodnocené hrany a cykly či negativní cykly, což jsou kružnice, pro které platí, že suma ohodnocení všech hran je záporná hodnota.

Fordův algoritmus je velmi podobný Dijkstrovu algoritmu – výstupem obou algoritmů je vektor vzdáleností od zdrojového vrcholu do všech ostatních vrcholů v grafu. Hlavní rozdíl, který je zároveň nejrelevantnější pro problematiku letového plánování, je skutečnost, že Fordův algoritmus – na rozdíl od Dijkstrova algoritmu – je aplikovatelný na grafy obsahující negativně ohodnocené hrany, což umožňuje jeho využití na širší řadu problémů. Fordův algoritmus je ovšem zároveň časově náročnější než Dijkstrův algoritmus s časovou náročností

$$\theta(|V| \cdot |E|) \tag{1}$$

Princip fungování Fordova algoritmu je založen na postupné relaxaci či uvolňování jednotlivých hran procházeného grafu ve všech iteracích. Jednotlivé kroky tohoto procesu mají následující podobu:

1. inicializace algoritmu – všem vrcholům V_i grafu $G = (V, E)$ označeným indexem $i = 1, \dots, n$ jako je přiřazeno ohodnocení $t_i = \infty$ a vrcholu v_i označenému indexem $i = 0$ (který je počátečním vrcholem hledané cesty) je přiřazeno ohodnocení $t_i = 0$

2. relaxace hran – pro každé dva vrcholy $v_i, v_j \in V_i$ je prováděna kontrola nerovnosti

   $$t_j - t_i > o(v_j, v_i)$$

   Pokud nerovnost platí, ohodnocení t_j je nahrazeno ohodnocením t'_j pro které platí

   $$t'_j = t_i + o(v_j, v_i)$$

   Pokud nerovnost neplatí, ohodnocení t_j zůstává beze změny.

3. detekce negativních cyklů – po relaxaci všech hran je doplňující iterace relaxace hran. V případě, kdy dojde k aktualizaci jednoho z ohodnocení t_j, obsahuje procházený graf negativní cyklus, a tudíž v něm nelze pomocí Fordova algoritmu nalézt minimální cestu.

4. výpis délky minimální cesty – v případě, že ve 3. kroku k aktualizaci nedojde, algoritmus proběhl v pořádku a lze získat délku minimální cesty $m^*(x, y)$ do vrcholu v_n, pro kterou platí

   $$t_n = \sum_{h \in m^*(x,y)} o(h) \tag{2}$$

Pseudokód algoritmu vypadá následovně [20]:

```
function bellmanFord(G, S)
  for each vertex V in G
    distance[V] <- infinite
      previous[V] <- NULL
  distance[S] <- 0

  for each vertex V in G
    for each edge (U,V) in G
      tempDistance <- distance[U] + edge_weight(U, V)
      if tempDistance < distance[V]
        distance[V] <- tempDistance
        previous[V] <- U

  for each edge (U,V) in G
    If distance[U] + edge_weight(U, V) < distance[V}
      Error: Negative Cycle Exists

  return distance[], previous[]
```

10 Další algoritmy pro hledání minimální cesty grafem

Vedle Fordova algoritmu je pro řešení problému hledání minimální cesty v grafu aplikovány také řada dalších algoritmů. Mezi tyto algoritmy se řadí zejména:

- Dijkstrův algoritmus,
- A* algoritmus.

Dijkstrův algoritmus

Dijkstrův algoritmus je jedním ze základních algoritmů pro hledání minimální cesty v grafu, který dokáže pro daný vrchol nalézt minimální cestu do všech ostatních vrcholů (popřípadě pouze do zvoleného koncového vrcholu, pokud je algoritmus doplněn o terminační podmínku).

Princip fungování algoritmu spočívá v iteračním procházení grafu přes jednotlivé vrcholy, kdy u každého vrchole dojde ke kontrole, zda přes s ním incidující hranou neexistuje kratší cesta do s ním sousedícího vrcholu.

A* algoritmus

A* algoritmus je další z algoritmů pro hledání minimálních cest v grafu, který v sobě spojuje původní mechanismy Dijkstrova algoritmu a heuristické metody pro vyhodnocování zvyšující celkovou efektivitu algoritmu. Samotný princip fungování algoritmu je založen na postupném (hrana po hraně) prodlužování cest ve stromu, jehož startem je počáteční vrchol hledané cesty. Při každé iteraci algoritmu dojde k vyhodnocení cenově (resp. distančně)

nejvýhodnější cesty, která je následně prodloužena. Vyhodnocování výhodnosti cest probíhá na základě nerovnosti

$$f(n) = g(n) + h(n) \tag{3}$$

Kde

- n je následující vrchol cesty,
- $g(n)$ je skutečné ohodnocení dosavadní délky (resp. ceny) cesty,
- $h(n)$ je hodnota heuristické funkce odhadující ohodnocení délky cesty z vrcholu n do vrcholu koncového,

K terminaci algoritmu dochází ve chvíli, kdy jedna z cest dosáhne koncového vrcholu. Klíčovým prvkem A* algoritmu je právě funkce $h(n)$, která je jeho hlavní výhodou vůči původnímu Dijkstrovu algoritmu a umožňuje svou modifikací využití na řadu nejrůznějších problémů.

Porovnání algoritmů pro hledání minimální cesty

Dijkstrův algoritmus je nejméně časově náročný. Existuje několik verzí s různou úrovní optimalizace časové náročnosti:

Původní algoritmus s časovou náročností

$$\theta((|V| + |E|) \cdot log|V|) \tag{4}$$

Maticová implementace původního algoritmu s časovou náročností

$$\theta(|V^2|) \tag{5}$$

Nižší časová náročnost je ovšem dosažena na úkor schopnosti algoritmu procházet grafy obsahující jak negativně ohodnocené cykly, tak negativně ohodnocené hrany. Výstupní informace algoritmu je vektor vzdáleností od zdrojového vrcholu do všech ostatních vrcholů.

Floydův algoritmus (v některých případech označován také jako Floyd-Warshallův algoritmus) je časově nejnáročnější s časovou náročností

$$\theta(|V^3|) \tag{6}$$

Hlavní výhodou Floydova algoritmu oproti Fordovu a Dijsktrovu algoritmu je skutečnost, že jeho výstupní informací je matice obsahující vzdálenosti z každého libovolného vrcholu do

každého jiného vrcholu. Algoritmus je zároveň schopen zpracovat grafy obsahující negativní hrany a upozornit na existenci negativních cyklů v grafu.

11 Modifikace Fordova algoritmu a jeho vstupních parametrů

Základní Fordův algoritmus (resp. zejména jeho vstupní parametry) je pro zohlednění vlivů větru nutno modifikovat. Těchto modifikací je dosaženo s pomocí několika nástrojů:

- poloviční versus sinus,
- Voroného diagramy,
- Delaného triangulace,
- zohlednění vlivů větru pro přepočet ohodnocení hran grafu.

11.1 Poloviční versus sinus

Vzorec pro poloviční versus sinus dokáže vypočítat ortodromu mezi dvěma body za předpokladu, že je známá jejich zeměpisná šířka a délka. Tento vzorec je velmi užitečný pro algoritmizaci vytvoření matice přímých vzdáleností v případě, že je uvažována fyzická vzdálenost mezi body a je třeba najít ohodnocení hran grafu.

Vzorec má následující podobu:

Nechť středový úhel mezi dvěma body je θ, pro kterou platí

$$\theta = \frac{d}{r} \tag{7}$$

Kde

- d je ortodroma mezi dvěma body na kružnici,
- r je poloměr kružnice.

Vzorec pro poloviční versus sinus pak dokáže jeho hodnotu získat pomocí hodnot zeměpisné šířky (λ) a délky (φ) a to následovně:

$$hav(\theta) = hav(\varphi_2 - \varphi_1) + \Big(1 - hav\big((\varphi_1 - \varphi_2)\big) - hav(\varphi_1 + \varphi_2)\Big) \cdot hav(\lambda_2 - \lambda_1) \tag{8}$$

Pro výpočet vzdálenosti *d* je následovně využit inverzní poloviční versus sinus, nebo alternativně arcus sinus:

$$d = r \cdot archav(h) = 2r \cdot arcsin\,(\sqrt{h}) \tag{9}$$

Kde

- h je rovno polovičnímu versus sinu θ (h = hav(θ)).

Odvozený vzorec vypadá následovně:

$$d = 2r \cdot arcsin\left(\sqrt{sin^2\left(\frac{\varphi_2 - \varphi_1}{2}\right) + cos\,\varphi_1 \cdot cos\,\varphi_2 \cdot sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right) \qquad (10)$$

Kde

- d je opět ortodroma mezi dvěma body na kružnici (respektive kouli),
- r je poloměr kružnice (resp. koule),
- λ_1 a λ_2 jsou hodnoty zeměpisné délky bodů 1 a 2,
- a φ_1 a φ_2 jsou hodnoty zeměpisné šířky bodů 1 a 2.

Nevýhodou vzorce je, že při použití v relaci se Zemí, se jedná o pouhou aproximaci, jelikož Země není dokonalá koule a její poloměr dosahuje hodnot 6 356,752 km na pólech až 6 378,136 km na rovníku. Zároveň, zakřivení zemského povrchu je na pólech (≈6 399,594 km) o 1% vyšší než na rovníku (≈6 335,137).

V rámci bakalářské práce je vzorec pro poloviční versus sinus (resp. odvozený vzorec pro výpočet vzdálenosti) aplikován pro zisk ohodnocení hran původního grafu, který nezohledňuje vlivy větru na délku (resp. dobu trvání) letu.

11.2 Voroného diagramy

Voroného diagram je způsob rozdělení plochy do oblastí, z nichž každá je v blízkosti skupiny daných objektů. V nejjednodušším případě jsou tyto objekty reprezentovány konečným počtem bodů (nazývanými jako jádra či generátory). Pro každé jádro pak existuje korespondující oblast nazývaná Voroného buňka, která je tvořena všemi body plochy, které jsou k danému bodu blíže než ke kterémukoliv dalšímu bodu.

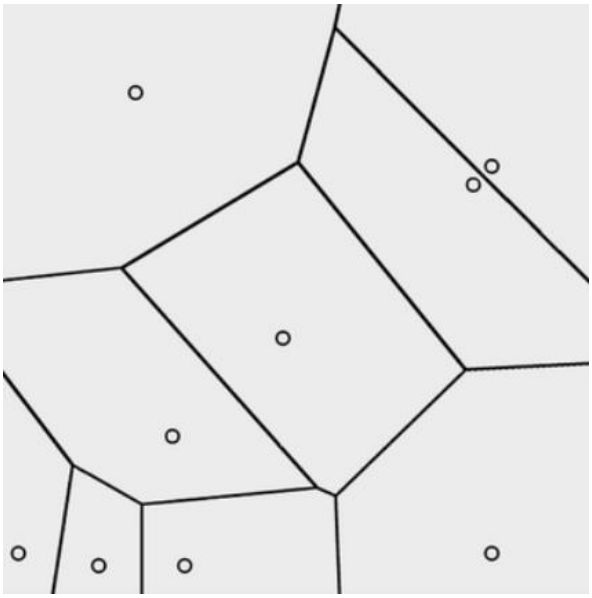
Formální definice Voroného diagramů je následující [24]:

Nechť X je metrický prostor se vzdálenostní funkcí d, K je sada indexů a $(P_k)k\varepsilon K$ je n-tice nenulových podmožin v prostoru X. Voroného buňka R_k asociovaná k jádru P_k je pak množina všech bodů v prostoru X, jejichž vzdálenost k P_k není větší než jejich vzdálenost ke každému ostatnímu jádru P_j, kde j je každý index mimo k – pokud

$$d(x, A) = inf\{d(x, a) | \, a \in A\} \qquad (11)$$

určuje vzdálenost mezi bodem x a podmnožinou A, pak

$$R_k = \{x \in X \mid d(x, P_k) \leq d(x, P_j)\} \quad pro\ všechna\ j \neq k \tag{12}$$

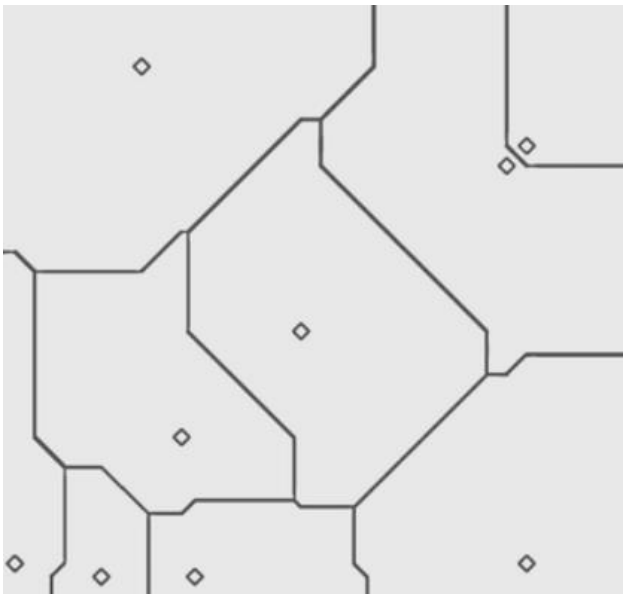


Obrázek 8 – Příklad Voroného diagramu s Euklidovskými vstupy [Zdroj: 25]

Modifikace Voroného diagramů

Existuje řada modifikací pro Voroného diagramy, které určitým způsobem upravují tvorbu oblastí okolo jejich jader. Mezi tyto modifikace patří například [24]:

- Mahalanobisovy Voroného diagramy – tato modifikace uvažuje kovarianční strukturu dat a měří vzdálenost jader od středu shluků všech jader v rámci prostoru,
- Manhattanovi Voroného diagramy – tato modifikace definuje vzdálenost mezi jednotlivými jádry jako součet absolutních hodnot rozdílů jejich souřadnic. V prostoru se lze pro tuto modifikaci pohybovat pouze vodorovně a svisle na mřížce,
- vážené Voroného diagramy – pro tuto modifikaci platí, že jednotlivým jádrům a jim přidruženým oblastem je funkcí určena multiplikativní (případně aditivní) váha, která ovlivní velikost vygenerované ho regionu.



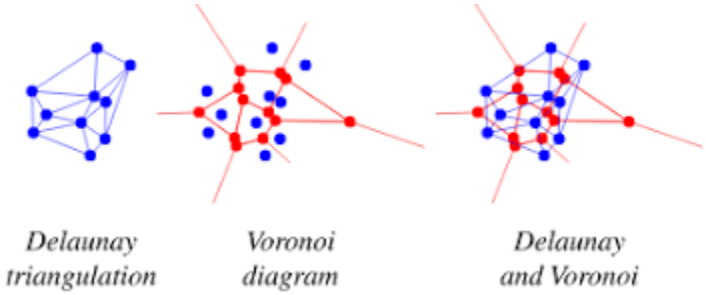
Obrázek 9 – Příklad Manhattanova Voroného diagramu [Zdroj: 25]

Jelikož rozložení rychlosti větrů nedodržuje pravidla kterékoliv z těchto modifikací, bude pro potřeby této práce aplikována Euklidovská verze.

11.3 Delaného triangulace

Delaného triangulace je jednou z metod geometrické analýzy pro rozdělení prostoru na trojúhelníky a zároveň představuje duální graf Voroného diagramů. Pro Delaného triangulaci platí, že musí splňovat tzv. Delaného podmínku, která udává, že žádný bod (zde vrchol grafu, resp. jádro oblasti Voroného diagramů) uvnitř nebo na hranici trojúhelníka nesmí mít kružnici opsanou, která by obsahovala jiný bod. [24]

Pro potřeby bakalářské práce je Delaného triangulace aplikována pro konstrukci vstupního grafu (resp. jeho hran) Fordova algoritmu.



Obrázek 10 Delaného triangulace a korespondující Voroného diagram [Zdroj: 21]

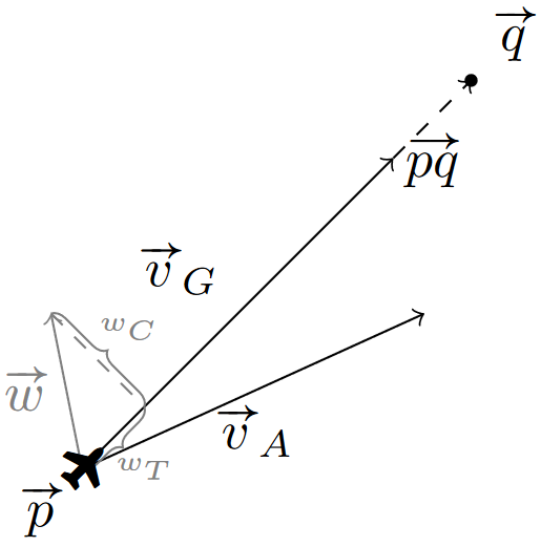
11.4 Modifikace algoritmu pro větrné podmínky

Pro modifikaci ohodnocení hran o vlivy povětrnostních podmínek (resp. rychlosti a směru větru v dané oblasti) byla použita následující rovnice.

$$t_{\vec{p},\vec{q}}(\vec{w}) = \frac{\| \overrightarrow{pq} \|}{\|\vec{v}_G\|} = \frac{\| \overrightarrow{pq} \|}{\omega cos\ \theta + \sqrt{h^2 - \omega^2 sin^2\ \theta}} \tag{13}$$

Jejíž princip graficky znázorňuje Obrázek 11 a kde

- $\vec{p}$ je počáteční bod cesty,
- $\vec{q}$ je koncový bod cesty,
- ω je velikost rychlosti větru (je-li uvažováno polární zobrazení vektoru větru $\vec{\omega} = (\omega, \theta)$),
- θ je úhel, který svírají vektory $\vec{\omega}$ a $\overrightarrow{pq}$ (je-li uvažováno polární zobrazení vektoru větru $\vec{\omega} = (\omega, \theta)$),
- h je skutečná vzdušná rychlost,
- $\vec{v}_G$ je rychlost letadla při zemi.



Obrázek 11 – Schéma vlivu vektoru větru na pohybový vektor letadla [Zdroj: 2]

12 Zdrojový kód

Příprava vstupních parametrů

Pro demonstrativní příklad byl vybrán seznam 28 největších letišť na území Německa, a to z důvodu vhodného počtu vzniklých vrcholů (28) a zároveň skutečnosti, že každé z těchto letišť pravidelně sdílí METAR data, která jsou pro další výpočty klíčová. Samotný graf byl

vytvořen za pomocí aplikace Delauného triangulace na Voroného diagram tvořen geografickými souřadnicemi vybraných letišť. Samotné vytvoření sítě není optimální a pro potřeby bakalářské práce slouží pouze jako mezikrok pro demonstraci fungování Fordova algoritmu a efektů rychlosti a směru větru na výslednou letovou trasu.

Skutečnou polohu těchto letišť znázorňuje Obrázek 12, zatímco jejich jmenný seznam společně s jejich korespondujícími kódy ICAO a IATA znázorňuje Tabulka 1.

Obrázek 12 – Mapa skutečných poloh letišť pro vzorový příklad [Zdroj: 18]

	{'icao': 'EDDF'	'iata': 'FRA'	'name': 'Frankfurt am Main International Airport'
1	{'icao': 'EDDM"	"iata': 'MUC"	"name': 'Munich International Airport"
2	{'icao': 'EDDB"	"iata': 'BER"	"name': 'Berlin Brandenburg Airport"
3	{'icao': 'EDDL"	"iata': 'DUS"	"name': 'Dusseldorf International Airport"
4	{'icao': 'EDDH"	"iata': 'HAM"	"name': 'Hamburg Airport"
5	{'icao': 'EDDK"	"iata': 'CGN"	"name': 'Cologne Bonn Airport"
6	{'icao': 'EDDS"	"iata': 'STR"	"name': 'Stuttgart Airport"
7	{'icao': 'EDDV"	"iata': 'HAJ"	"name': 'Hannover Airport"
8	{'icao': 'EDLW"	"iata': 'DTM"	"name': 'Dortmund Airport"
9	{'icao': 'EDDN"	"iata': 'NUE"	"name': 'Nuremberg Airport"
10	{'icao': 'EDJA"	"iata': 'FMM"	"name': 'Memmingen Allgau Airport"
11	{'icao': 'EDFH"	"iata': 'HHN"	"name': 'Frankfurt-Hahn Airport"
12	{'icao': 'EDDP"	"iata': 'LEJ"	"name': 'Leipzig Halle Airport"
13	{'icao': 'EDDW"	"iata': 'BRE"	"name': 'Bremen Airport"
14	{'icao': 'EDSB"	"iata': 'FKB"	"name': 'Karlsruhe Baden-Baden Airport"
15	{'icao': 'EDLV"	"iata': 'NRN"	"name': 'Niederrhein Airport"
16	{'icao': 'EDDG"	"iata': 'FMO"	"name': 'Munster Osnabruck Airport"
17	{'icao': 'EDDC"	"iata': 'DRS"	"name': 'Dresden Airport"
18	{'icao': 'EDDR"	"iata': 'SCN"	"name': 'Saarbrucken Airport"
19	{'icao': 'EDLP"	"iata': 'PAD"	"name': 'Paderborn Lippstadt Airport"
20	{'icao': 'EDNY"	"iata': 'FDH"	"name': 'Friedrichshafen Airport"
21	{'icao': 'EDXW"	"iata': 'GWT"	"name': 'Westerland Sylt Airport"
22	{'icao': 'EDDE"	"iata': 'ERF"	"name': 'Erfurt Airport"
23	{'icao': 'EDLN"	"iata': 'MGL"	"name': 'Monchengladbach Airport"
24	{'icao': 'EDVK"	"iata': 'KSF"	"name': 'Kassel-Calden Airport"
25	{'icao': 'EDVE"	"iata': 'BWE"	"name': 'Braunschweig Wolfsburg Airport"
26	{'icao': 'ETNL"	"iata': 'RLG"	"name': 'Rostock-Laage Airport"
27	{'icao': 'EDHL"	"iata': 'LBC"	"name': 'Lubeck Blankensee Airport"

Tabulka 1 – Seznam vrcholů (zde letišť a jejich ICAO a IATA kódů) zpracovávaného grafu [autor]

Matlab script pro izolování kódů ICAO

Tento Matlab Livescript kód obsahuje několik příkazů pro manipulaci s daty a soubory, jehož konečným výsledkem je izolace vektoru (sloupce) obsahující kódy ICAO jednotlivých letišť. Po vyčištění pracovního prostoru jsou data načtena z tabulky uložené v souboru "letiste2.txt". Poté je jeden sloupec tabulky převeden na řetězce (string). Následuje příprava dat pro export, kdy jsou hodnoty ze sloupce spojeny do jednoho řetězce s oddělovačem čárka. Vytvoří se soubor "DataArray.txt" pro uložení výstupních dat, do kterého je zapsán připravený řetězec. Na závěr se soubor uzavře.

```
clc
clear
```

```
close all



T = readtable("letiste2.txt","Delimiter","\t",ReadVariableNames=false)


T.Var3 = string(T.Var3)


preparationforexport = strjoin(T.Var3,",")


savefilename = "DataArray.txt"
fileID2 = fopen(savefilename,'w');
fprintf(fileID2,preparationforexport);
fclose(fileID2);
```

Python script pro stažení současných METAR dat z vybraných letišť

Tento Python kód využívá knihovnu `pyflightdata` pro získávání vybraných leteckých dat (v tomto případě zprávy METAR pro dané letiště) a následně je ukládá do .txt souboru. Jednotlivé jeho sekce fungují následovně:

- importování modulů: kód začíná importováním modulů `FlightData` z knihovny `pyflightdata` a `time`,
- inicializace objektu `f`: vytváří se instance třídy `FlightData` a přiřazuje se do proměnné `f`,
- přihlášení: zavoláním metody `login` na objektu `f` se přihlašuje uživatel pomocí zadaného e-mailu a hesla,
- načtení dat: soubor "DataArray.txt" je otevřen a jeho obsah je přečten a uložen do proměnné `lines`, načež je tento řetězec rozdělen pomocí čárky a výsledkem je seznam hodnot přiřazený do proměnné `newlines`,
- cyklus pro získání dat: vytváří se prázdný seznam `M`, načež sepro každý prvek `i` v `newlines` zavolá metoda `get_airport_metars` na objektu `f` s aktuálním prvkem jako argumentem; výsledek je přidán na konec seznamu `M` a tiskne se hodnota `i` pro sledování průběhu a pomocí `time.sleep(3)` se čeká 3 sekundy mezi každým voláním metody (tato funkce je nutná, jelikož by v případě její absence server FlightRadar rozpoznal, že dotazy pokládá robot a následně je zamítl),
- odhlášení: metoda `logout` je volána na objektu `f` pro odhlášení,

- zápis do souboru:sSoubor "MetarGermany.txt" je otevřen pro zápis a následuje cyklus `while`, který projde všechny prvky v seznamu `M`, načež je každý prvek převeden na řetězec a zapsán do souboru; za každým řádkem je přidána nová řádka; na konci je soubor uzavřen.

```
from pyflightdata import FlightData
import time


f = FlightData()
f.login("FakeFaker69@seznam.cz", "bpracek617")


lines = []
with open("DataArray.txt") as fil:
    lines = fil.read()
newlines = lines.split(",")


M = []
for i in range(len(newlines)):
    a = f.get_airport_metars(newlines[i])
    M.append(a)
    print(i)
    time.sleep(3)
f.logout()
print(M)


file = open("MetarGermany.txt", "w")
i = 0
while i < len(M):
    b = str(M[i])
    file.write(b)
    file.write("\n")
    i = i + 1
file.close()
```

Python script pro stažení souřadnic vybraných letišť

Tento kód importuje moduly `pyflightdata` a `airportsdata` a používá je k provádění operací souvisejících s letišti a letovými daty – v tomto případě se jedná o informace o daných letištích (zejména jejich zeměpisné souřadnice). Následně načítá data ze souboru "DataArray.txt" a provádí jejich zpracování.

Modul `airportsdata` slouží k načítání informací o letištích na základě ICAO kódů. První část kódu načítá tato data do proměnné `airports` pomocí funkce `load()` z modulu `airportsdata`.

Dále se přistupuje k souboru "DataArray.txt" pomocí příkazu `with open("DataArray.txt") as fil`. Obsah souboru je načten do proměnné `lines` funkcí `read()`. Poté je řetězec rozdělen na seznam pomocí metody `split(",")`, kde oddělovačem je čárka. Výsledkem je seznam hodnot uložený v proměnné `newlines`.

Následuje vytvoření prázdného seznamu `M`. Ve smyčce `for` procházíme jednotlivé prvky seznamu `newlines` a pomocí ICAO kódu získáváme informace o příslušném letišti z proměnné `airports`. Získaná data jsou přidávána do seznamu `M` pomocí metody `append(a)`. Současně je v každém kroku vypsána hodnota `i` pomocí příkazu `print(i)`.

Poté je otevřen soubor "CoordsGermany.txt" v režimu zápisu (`"w"`). Následuje smyčka `while`, která postupně prochází prvky seznamu `M`. Každý prvek je převeden na řetězec pomocí funkce `str()` a zapsán na samostatný řádek do souboru pomocí metody `write()`. Na konci každého řádku je přidána nová řádka (`"\n"`). Smyčka pokračuje, dokud nejsou zpracovány všechny prvky seznamu `M`. Nakonec je soubor uzavřen pomocí příkazu `close()`.

```
from pyflightdata import FlightData
import airportsdata
import time


airports = airportsdata.load("ICAO")


lines = []
with open("DataArray.txt") as fil:
    lines = fil.read()
newlines = lines.split(",")


M = []
for i in range(len(newlines)):
    a = airports[newlines[i]]
    M.append(a)
    print(i)


print(M)


file = open("CoordsGermany.txt", "w")
i = 0
while i < len(M):
```

```
    b = str(M[i])
    file.write(b)
    file.write("\n")
    i = i + 1
file.close()
```

Matlab script pro vytvoření vektorů rychlosti a směru větru

Kód slouží ke zpracování dat z meteorologických informací, které jsou uloženy v souboru "MetarGermany.txt". Zde je popis jednotlivých částí kódu:

- příprava prostředí: kód začíná vymazáním proměnných, vyčištěním příkazového řádku a zavřením všech otevřených objektů,
- načtení dat: kód načítá data ze souboru "MetarGermany.txt" a ukládá je do tabulky `N`, načež se vytváří proměnná `plus` pro opravení načtených dat (část dat byla uložena do záhlaví tabulky),
- příprava proměnných: z tabulky `N` se extrahuje sloupec obsahující data o rychlosti a směru větru a ukládá se do proměnné `P`,
- rozdělení vektoru `P` na dva distinktní vektory - rychlost a směr větru: proměnná `P` je převedena z buňkového pole na matici pomocí funkce `cell2mat()`; poté se rozděluje na vektory pro rychlost a směr větru - rychlost je uložena do proměnné `W` a směr je uložen do proměnné `Z`,
- ošetření případů se zanedbatelnou rychlostí: algoritmus prochází jednotlivé hodnoty směru větru ve for cyklu a kontroluje, zda hodnota začíná písmenem "V". Pokud ano, do proměnné `D` se přiřadí hodnota "V" z důvodu následného snadnějšího rozlišení datových typů; v opačném případě se provede převod čísla z řetězce na číselnou hodnotu a tato hodnota se uloží do proměnné `D`,
- převod rychlosti větru na číselnou hodnotu: prochází se jednotlivé hodnoty rychlosti větru ve smyčce `for` a provádí se převod čísla z řetězce na číselnou hodnotu. Tyto hodnoty se ukládají do vektoru `S`.

```
clear
clc
close all
```

```
N = readtable("MetarGermany.txt", "Delimiter","
","VariableNamingRule","preserve")
plus = N.Properties.VariableNames


P = N.("03008KT");
P2 = plus(1,4)
P(length(P) + 1) = P2
P = cell2mat(P);
Z = P(:,1:3);
W = P(:,4:5);


for i = 1:length(Z)
    if Z(i,1) == "V"
        D(i,1) = "V"
    else
        D(i,1) = str2double(Z(i,1:3));
    end
end


for i = 1:length(W)
    S(i,1) = str2double(W(i,1:2));
end
```

Matlab script pro vytvoření Voroného diagramu a Delaného triangulace

Hlavním účelem kódu je konstrukce Voroného diagramů a Delaného triangulace a následná příprava vstupních dat pro Fordův algoritmus. Stručný popis jednotlivých částí kódu vypadá následovně:

- příprava prostředí: kód začíná vymazáním proměnných, vyčištěním příkazového okna a zavřením všech otevřených objektů,
- načtení meteorologických dat: kód načítá data ze souboru "MetarGermany.txt" a ukládá je do tabulky `Metar`, načež je vytvořena proměnná `plus` pro opravení načtených dat (část dat byla uložena do záhlaví tabulky),
- zpracování rychlosti a směru větru: z dat v tabulce `Metar` se extrahují rychlost a směr větru; rychlost je převedena na kilometry za hodinu a směr je zpracován do číselného formátu (stupně a minuty),
- načtení souřadnic letišť: kód načítá data (souřadnice) ze souboru "CoordsGermany.txt" a ukládá je do tabulky `M`; opět se vytváří proměnná `plus` pro opravení načtených dat,

- příprava vektoru ICAO kódů: z tabulky `M` se extrahuje sloupec s ICAO kódy letišť a připravuje se vektor `ICAO`,
- příprava vektoru zeměpisné šířky: z tabulky `M` se extrahuje sloupec s hodnotami zeměpisné šířky a připravuje se vektor `LAT`,
- příprava vektoru zeměpisné délky: z tabulky `M` se extrahuje sloupec s hodnotami zeměpisné délky a připravuje se vektor `LON`,
- vytvoření matice přímých vzdáleností: vytváří se matice `Matice` a `MaticeAZ` pro ukládání přímých vzdáleností a azimutů mezi hranami spojujícími vybrané letiště – vnitřními cykly se počítají vzdálenosti pomocí funkce `distance`, která je součástí balíčku „Mapping toolbox“ a jejíž základní princip funguje na základě vzorce pro poloviční versus sinus, a převádějí na kilometry,
- vykreslení Voroného diagramu a Delaunay triangulace: kód vytváří grafické zobrazení letišť pomocí funkcí `voronoi` a `delaunayTriangulation`; dále jsou získány hrany triangulace,
- výpočet doby letu s vlivem větru: kód vytváří matici `MaticeCasWind`, ve které jsou vypočítány doby letu mezi letišti s ohledem na rychlost a směr větru,
- výpočet doby letu bez vlivu větru: kód vytváří matici `MaticeCas`, ve které jsou vypočítány doby letu mezi letišti bez vlivu větru; výpočty jsou založeny pouze na přímých vzdálenostech a rychlosti letadla,
- vykreslení grafů: kód vytváří grafy z vytvořených matic a zobrazuje je; v grafu jsou hrany označeny váhami odpovídajícími době letu,
- příprava matic pro Fordův algoritmus: v maticích `MaticeCasWind`, `MaticeCas` a `Matice` se nulují hodnoty pro neplatné hrany a nahrazují se nekonečnem (inf).

```
clear
clc
close all


%Nacteni zprav metar a vyhodnoceni rychlosti a smeru vetru


Metar = readtable("MetarGermany.txt", "Delimiter","
","VariableNamingRule","preserve")
plus = Metar.Properties.VariableNames
```

```
RepairMet = Metar.("03008KT");
RepairMetHead = plus(1,4)
RepairMet(length(RepairMet) + 1) = RepairMetHead
RepairMet = cell2mat(RepairMet);
WindDirTest = RepairMet(:,1:3);
WindVelTest = RepairMet(:,4:5);


for i = 1:length(WindDirTest)
    if WindDirTest(i,1) == "V"                      %ošetření případů, kdy je
rychlost zanedbatelná
        Dir(i,1) = "V";
    else
        Dir(i,1) = str2double(WindDirTest(i,1:3));
    end
end


for i = 1:length(WindVelTest)
    Vel(i,1) = str2double(WindVelTest(i,1:2));
end


Vel = 1.852*Vel             %převod na km%h
Dir(isnan(Dir)) = 0


%Nacteni souradnic letist


M = readtable("CoordsGermany.txt", "Delimiter",","  ...
    , "VariableNamingRule","preserve")
plus = M.Properties.VariableNames;


%vector kodu ICAO


ICAO = M.("icao: 'EDDF'");
ICAO2 = plus(1,2);
ICAO(length(ICAO) + 1) = ICAO2;
ICAO3 = cell2mat(ICAO);
ICAO = ICAO3(:,10:12);


%vector zem. sirky


LAT = M.("'lat': 50.0264015198");
LAT2 = plus(1,8);
LAT(length(LAT) + 1) = LAT2
```

```
LAT3 = cell2mat(LAT);
LAT = LAT3(:,8:20)
LATFin = zeros(28,1);


for i = 1:length(LAT)
    LATFin(i,1) = str2double(LAT(i,1:13));
end
%LATF(1,1)                                    kontrola


%vektor zem. delky


LON = M.("'lon': 8.54312992100");
LON2 = plus(1,9);
LON(length(LON) + 1) = LON2;


LON3 = cell2mat(LON);
LON = LON3(:,8:20)
LONFin = zeros(28,1);
for i = 1:length(LON)
    LONFin(i,1) = str2double(LON(i,1:13));
end
%LONFin(1,1)                                 kontrola


%matice primych vzdalenosti


Matice = [];
MaticeAZ = [];
velikost2 = length(LATFin)


for i = 1:velikost2
    for j = 1:velikost2
        [Matice(i, j),MaticeAZ(i, j)] = distance(LONFin(i), ...
            LATFin(i),LONFin(j),LATFin(j));
    end
end


Matice = deg2km(Matice)
MaticeAZ                          % hodnoty pro kompletí graf - nutno
                                  % převést na delaného triangulaci
%plot(digraph(MaticeAZ))


%voroneho diagram + delauneho triangulace
```

```
figure
voronoi(LONFin,LATFin)
plabels = ICAO


hold on
Hpl = text(LONFin, LATFin, plabels, 'FontWeight','bold',...
        'HorizontalAlignment','center', ...
        'BackgroundColor', 'none');
hold off


dt = delaunayTriangulation(LONFin, LATFin);
hold on
triplot(dt,'-r');
hold off


E1 = edges(dt)
E2 = [];
E2(:,1) = E1(:,2);
E2(:,2) = E1(:,1);
E = [E1; E2]
%plot(digraph(E(:,1),E(:,2)))


%Tvorba matice primych vzdalenosti


zero_mask = ones(size(Matice));
zero_mask(sub2ind(size(Matice), E(:,1), E(:,2))) = 0;


Matice(zero_mask == 1) = 0


zero_mask = ones(size(MaticeAZ));
zero_mask(sub2ind(size(MaticeAZ), E(:,1), E(:,2))) = 0;


MaticeAZ(zero_mask == 1) = 0




plot(graph(Matice,"upper"), "EdgeLabel", graph(Matice).Edges.Weight,
"NodeLabel", string(ICAO), EdgeFontSize = 5)


%doba trvani letu s efekty vetru


h = 550;                    % km/h = - kts
```

```
MaticeCasWind = [];


for i = 1:velikost2
    for j = 1:velikost2
        MaticeCasWind(i, j) = ((Matice(i, j) / (Vel(i) * cos(MaticeAZ(i, j) -
Dir(i)) + sqrt((h^2) - (Vel(i))^2 * (sin(MaticeAZ(i, j) - Dir(i)))^2))) +
(Matice(i, j) / (Vel(j) * cos(MaticeAZ(i, j) - Dir(j)) + sqrt((h^2) -
(Vel(j))^2 * (sin(MaticeAZ(i, j) - Dir(j)))^2)))) / 2;
    end
end


MaticeCasWind           %hodiny


%doba trvani letu bez efektu vetru


MaticeCas = Matice ./ h


plot(digraph(MaticeCasWind), "EdgeLabel", digraph(MaticeCasWind).Edges.Weight,
"NodeLabel", string(ICAO), EdgeFontSize = 5)
plot(digraph(MaticeCas), "EdgeLabel", digraph(MaticeCas).Edges.Weight,
"NodeLabel", string(ICAO), EdgeFontSize = 5)


%prepis matic pro forda
MaticeCasWind(MaticeCasWind == 0) = inf;
MaticeCas(MaticeCas == 0) = inf;
Matice(Matice == 0) = inf;
```

Matlab script pro výpočet Fordova algoritmu pro hodnoty bez vlivu větru

Kód v MATLAB Livescriptu implementuje algoritmus Dijkstrova pro hledání minimální cesty v grafu. Zde je stručný popis jednotlivých částí:

- inicializace proměnných: nastavují se počáteční hodnoty proměnných `start`, `PocetVrcholu` a `ohodnoceni`. `ohodnoceni` je vektor, jehož prvky po ukončení chodu kódu udávají hodnoty délky minimální cesty do vrcholu s korespondujícím indexem,
- příprava dat pro algoritmus: vytvářejí se prázdné pole `PocVrcholy`, `KoncVrcholy` a `HranoveHodnnotyBezVetru`, které budou sloužit k ukládání informací o hranách grafu a projíždí se matice `MaticeCas`, což je matice přímých vzdáleností daného grafu a získávají se z ní potřebné vstupní hodnoty pro Fordův algoritmus,

- hledání minimálních cest: prochází se graf a aktualizují se ohodnocení vrcholů a ukládají se předchůdci na minimální cestě,
- výpis výsledků: Vypisují se vzdálenosti od počátečního vrcholu ke všem ostatním vrcholům,
- zobrazení cesty v grafu: je vytvořen se graf `G` s pomocí funkcí `digraph` a `plot`; poté se zobrazuje minimální cesta mezi vybranými vrcholy v grafu pomocí funkce `highlight`,
- výpis ohodnocení a předchůdců: vypisují se vektor ohodnocení vrcholů a vektor předchůdců.

```
start = 1;
PocetVrcholu = height(MaticeCas);
ohodnoceni = zeros(PocetVrcholu,1);
ohodnoceni(1:PocetVrcholu) = inf;
ohodnoceni(start) = 0;
ohodnoceni = transpose(ohodnoceni);



PocVrcholy = [];
KoncVrcholy = [];


HranoveHodnnotyBezVetru = [];
ohodnoceniBezVetru = ohodnoceni


for i = 1:PocetVrcholu
    for j = 1:PocetVrcholu
        if MaticeCas(i, j) ~= inf && i ~= j
            PocVrcholy = [PocVrcholy, i];
            KoncVrcholy = [KoncVrcholy, j];
            HranoveHodnnotyBezVetru = [HranoveHodnnotyBezVetru, MaticeCas(i,
j)];
        end
    end
end


PocVrcholy
KoncVrcholy
HranoveHodnnotyBezVetru
PocetHran = length(PocVrcholy)
n = length(MaticeCas);
predchudceBezVetru = zeros(1, n);
```

```
for i = 1:n-1
    for j = 1:numel(PocVrcholy)
        if ohodnoceniBezVetru(PocVrcholy(j)) + HranoveHodnnotyBezVetru(j) <
ohodnoceniBezVetru(KoncVrcholy(j))
            ohodnoceniBezVetru(KoncVrcholy(j)) =
ohodnoceniBezVetru(PocVrcholy(j)) + HranoveHodnnotyBezVetru(j);
            predchudceBezVetru(KoncVrcholy(j)) = PocVrcholy(j);
        end
    end
end


% Výpis výsledků
for i = 1:n
    fprintf('Vzdálenost od vrcholu 1 do vrcholu %d: %d\n', i,
ohodnoceniBezVetru(i));
end


KoncovyVrchol = 4; % Změňte tento číselný index na požadovaný koncový vrchol
CestaBezVetru = [KoncovyVrchol];
while KoncovyVrchol ~= 1
    KoncovyVrchol = predchudceBezVetru(KoncovyVrchol);
    CestaBezVetru = [KoncovyVrchol CestaBezVetru];
end


% Zobrazení cesty v grafu
G = digraph(PocVrcholy, KoncVrcholy, HranoveHodnnotyBezVetru);
plot(G,"Layout","force","EdgeLabel",G.Edges.Weight, EdgeFontSize = 5,
NodeLabel=string(ICAO));
Graf = plot(G,"Layout","force","EdgeLabel",G.Edges.Weight, EdgeFontSize = 5,
NodeLabel=string(ICAO));


highlight(Graf, CestaBezVetru, 'EdgeColor', 'r', 'LineWidth', 2);


ohodnoceniBezVetru
predchudceBezVetru
```

Matlab script pro výpočet Fordova algoritmu pro hodnoty zohledňující vlivy větru

Kód funguje na naprosto totožném principu jako kód v podkapitole Matlab script pro výpočet Fordova algoritmu pro hodnoty bez vlivu větru s tím rozdílem, že místo matice přímých vzdáleností obsahující hodnoty trvání cesty nezohledňující vlivy větru je použita matice přímých vzdáleností obsahující hodnoty trvání cesty zohledňující vlivy větru.

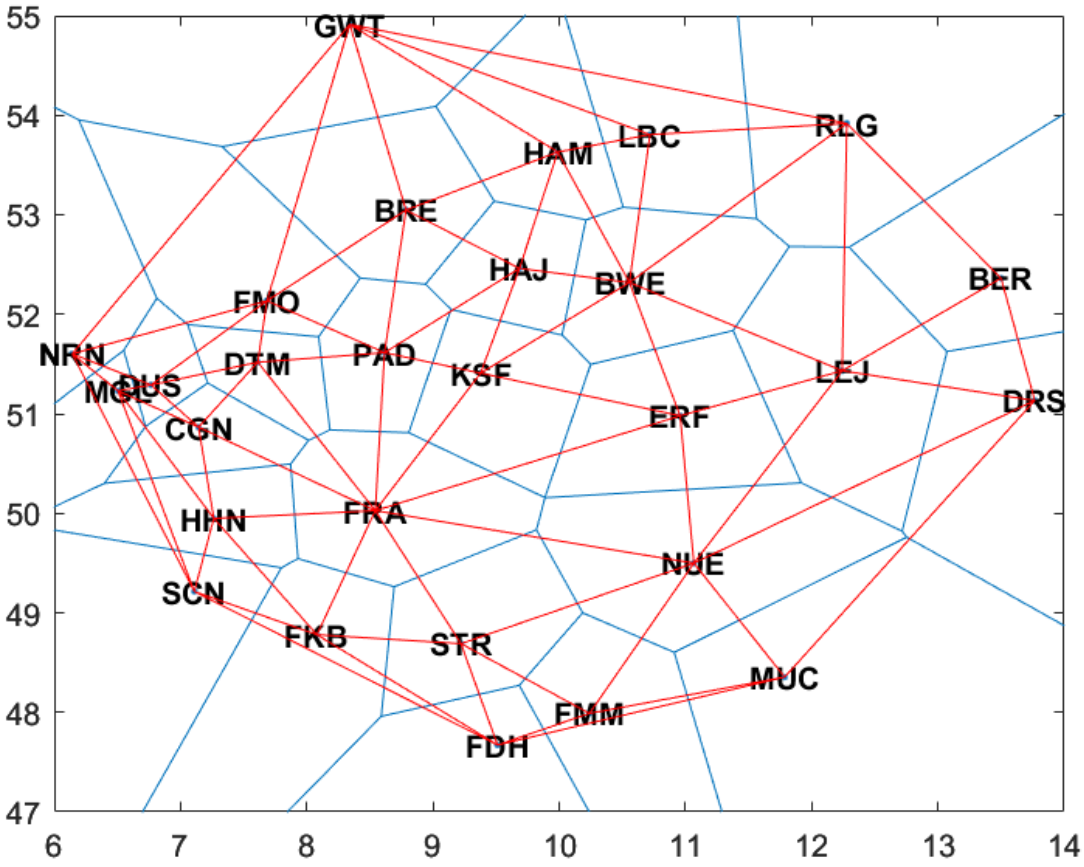
```
start = 1;
```

```
PocetVrcholu = height(MaticeCas);
ohodnoceni = zeros(PocetVrcholu,1);
ohodnoceni(1:PocetVrcholu) = inf;
ohodnoceni(start) = 0;
ohodnoceni = transpose(ohodnoceni);




PocVrcholy = [];
KoncVrcholy = [];

HranoveHodnnotySVetrem = [];
ohodnoceniSVetrem = ohodnoceni


for i = 1:PocetVrcholu
    for j = 1:PocetVrcholu
        if MaticeCasWind(i, j) ~= inf && i ~= j
            PocVrcholy = [PocVrcholy, i];
            KoncVrcholy = [KoncVrcholy, j];
            HranoveHodnnotySVetrem = [HranoveHodnnotySVetrem, MaticeCasWind(i,
j)];
        end
    end
end


PocVrcholy
KoncVrcholy
HranoveHodnnotySVetrem
PocetHran = length(PocVrcholy)
n = length(MaticeCasWind);
predchudceSVetrem = zeros(1, n);


for i = 1:n-1
    for j = 1:numel(PocVrcholy)
        if ohodnoceniSVetrem(PocVrcholy(j)) + HranoveHodnnotySVetrem(j) <
ohodnoceniSVetrem(KoncVrcholy(j))
            ohodnoceniSVetrem(KoncVrcholy(j)) =
ohodnoceniSVetrem(PocVrcholy(j)) + HranoveHodnnotySVetrem(j);
            predchudceSVetrem(KoncVrcholy(j)) = PocVrcholy(j);
        end
    end
end


% Výpis výsledků
for i = 1:n
    fprintf('Vzdálenost od vrcholu 1 do vrcholu %d: %d\n', i,
ohodnoceniSVetrem(i));
end
```

```
KoncovyVrchol = 4; % Změňte tento číselný index na požadovaný koncový vrchol
CestaSVetrem = [KoncovyVrchol];
while KoncovyVrchol ~= 1
    KoncovyVrchol = predchudceSVetrem(KoncovyVrchol);
    CestaSVetrem = [KoncovyVrchol CestaSVetrem];
end


% Zobrazení cesty v grafu
G = digraph(PocVrcholy, KoncVrcholy, HranoveHodnnotySVetrem);
plot(G,"Layout","force","EdgeLabel",G.Edges.Weight, EdgeFontSize = 5,
NodeLabel=string(ICAO));
Graf = plot(G,"Layout","force","EdgeLabel",G.Edges.Weight, EdgeFontSize = 5,
NodeLabel=string(ICAO));


highlight(Graf, CestaSVetrem, 'EdgeColor', 'r', 'LineWidth', 2);


ohodnoceniSVetrem
predchudceSVetrem
```

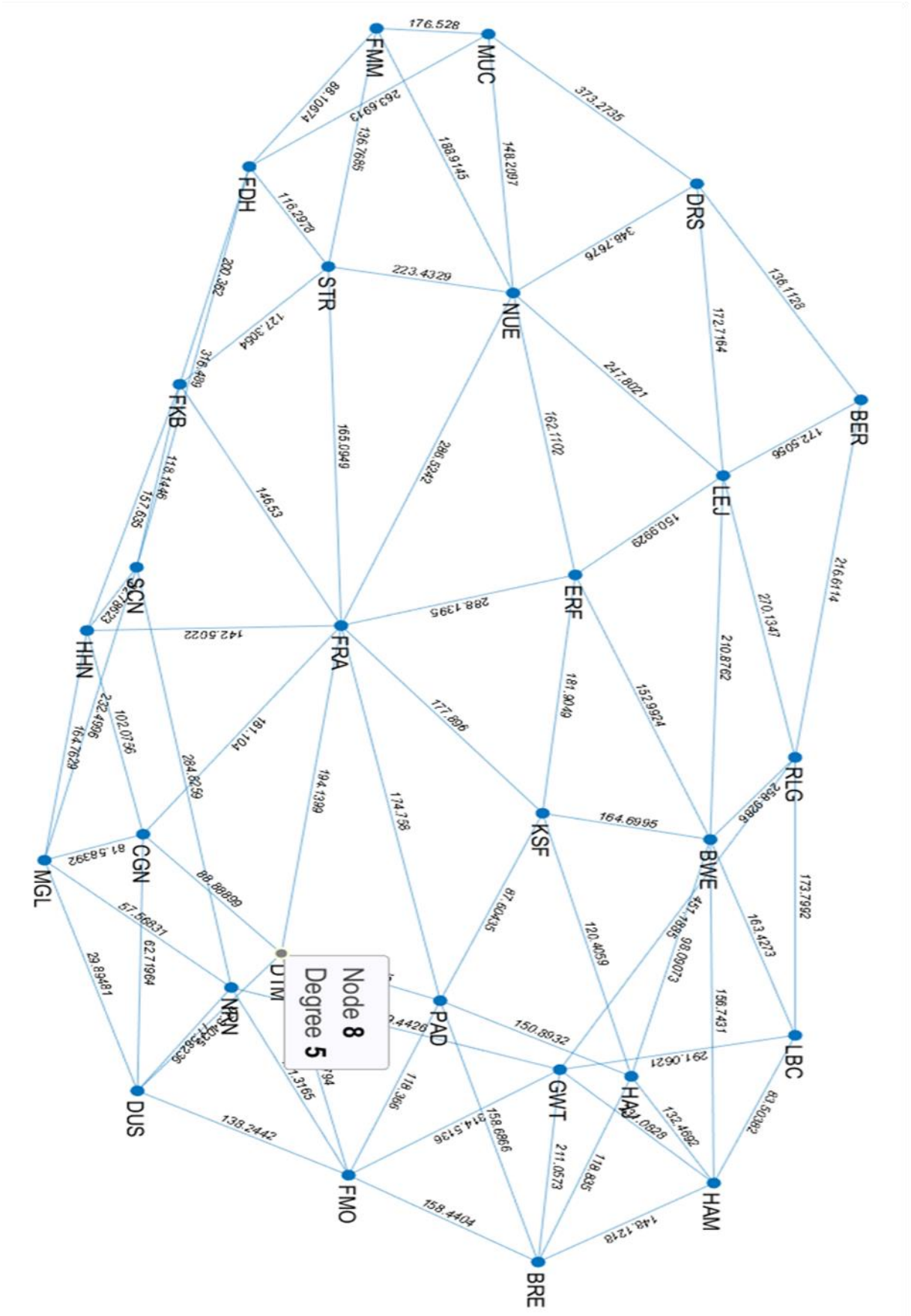
13 Výstupy

Mezivýstupem algoritmu je síť tvořena Voroného diagramem a Delaného triangulací souřadnic zpracovávaných letišť. Červené linie zde představují hrany výsledného grafu (vytvořené právě Delaného triangulací) a modré linie jsou výstup Voroného diagramů rozdělující graf na podoblasti, kde pro každý region existuje jeho jádro (letiště, které je zároveň vrcholem grafu) a korespondující zpráva METAR a z ní získané hodnoty rychlosti a směru větru.



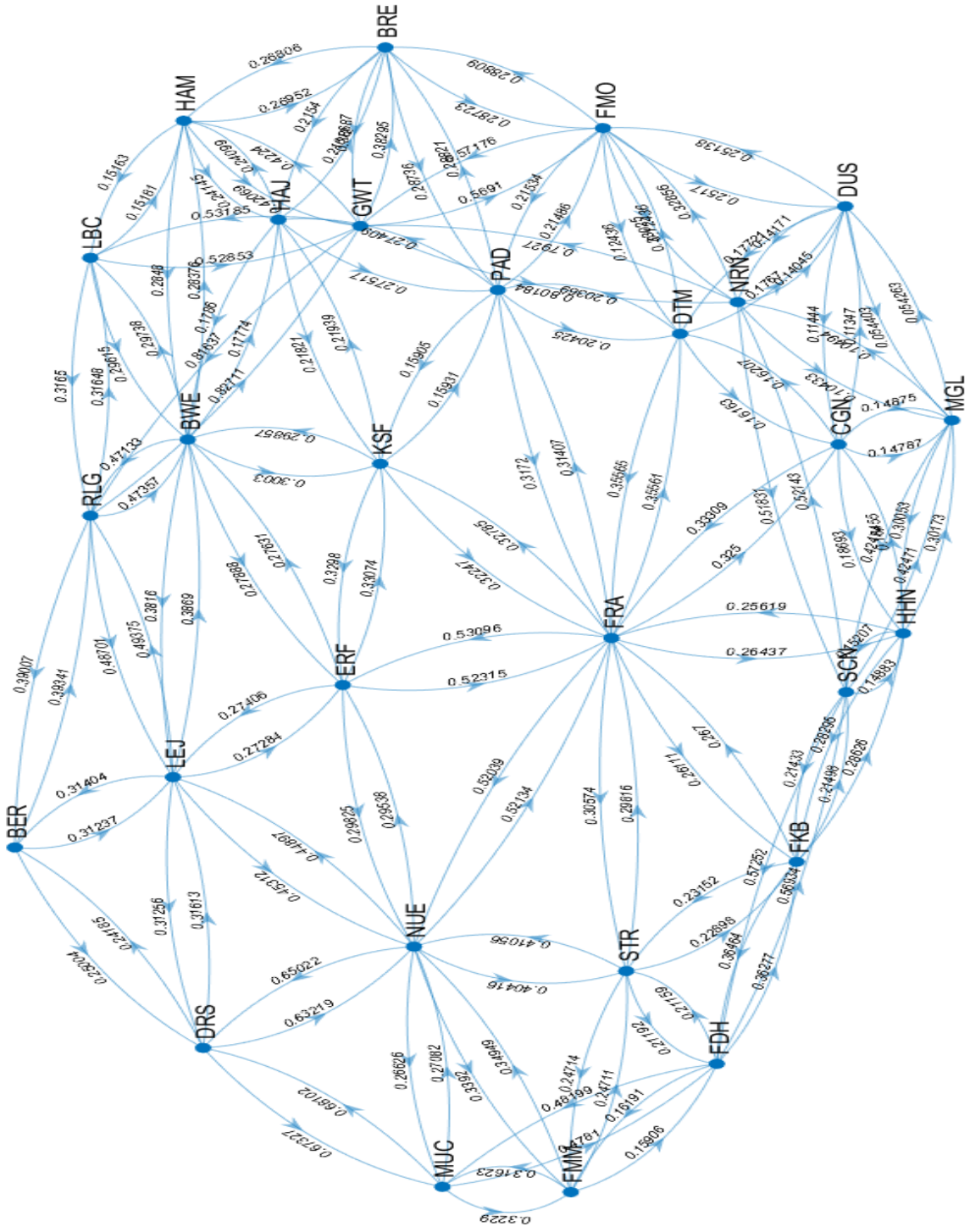
Obrázek 13 – Voroného diagram a Delaného triangulace letového prostoru Německa [autor]

Za využití vzorce pro poloviční versus sinus souřadnic jednotlivých letišť získaných kódem lze jednotlivým hranám přiřadit také jejich skutečnou délku v kilometrech. Takto vzniklý graf znázorňuje Obrázek 14.



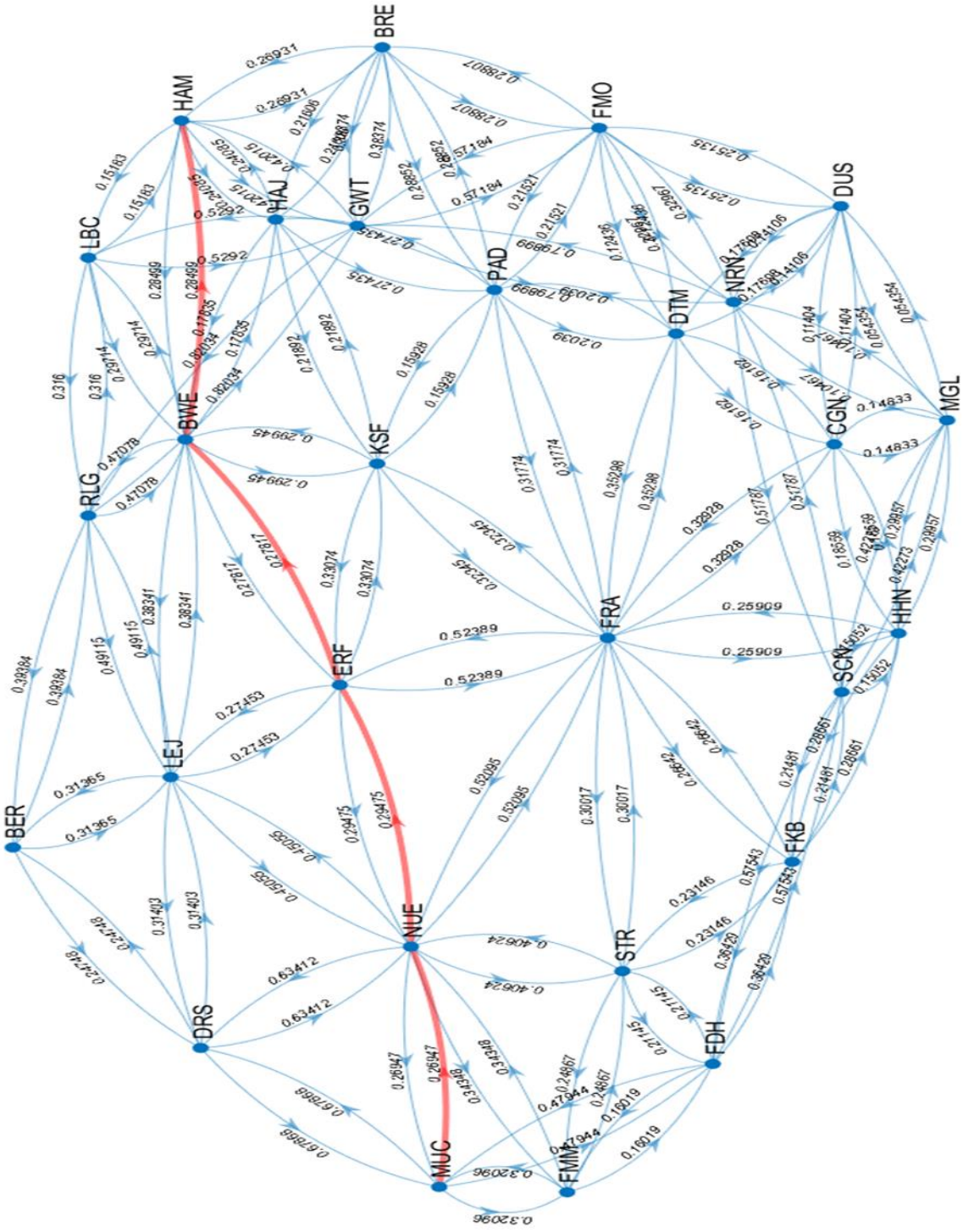
Obrázek 14 – Graf zkoumané oblasti doplněný o skutečné délky hran [autor]

Tento graf je následně za aplikace vzorce pro zohlednění vlivů rychlosti a směru větru transformován na digraf, jehož hrany jsou ohodnoceny délkou trvání přeletu dané hrany a který znázorňuje Obrázek 15.



Obrázek 15 – digraf zkoumané oblasti již doplněn o vlivy rychlosti a směru větru [autor]

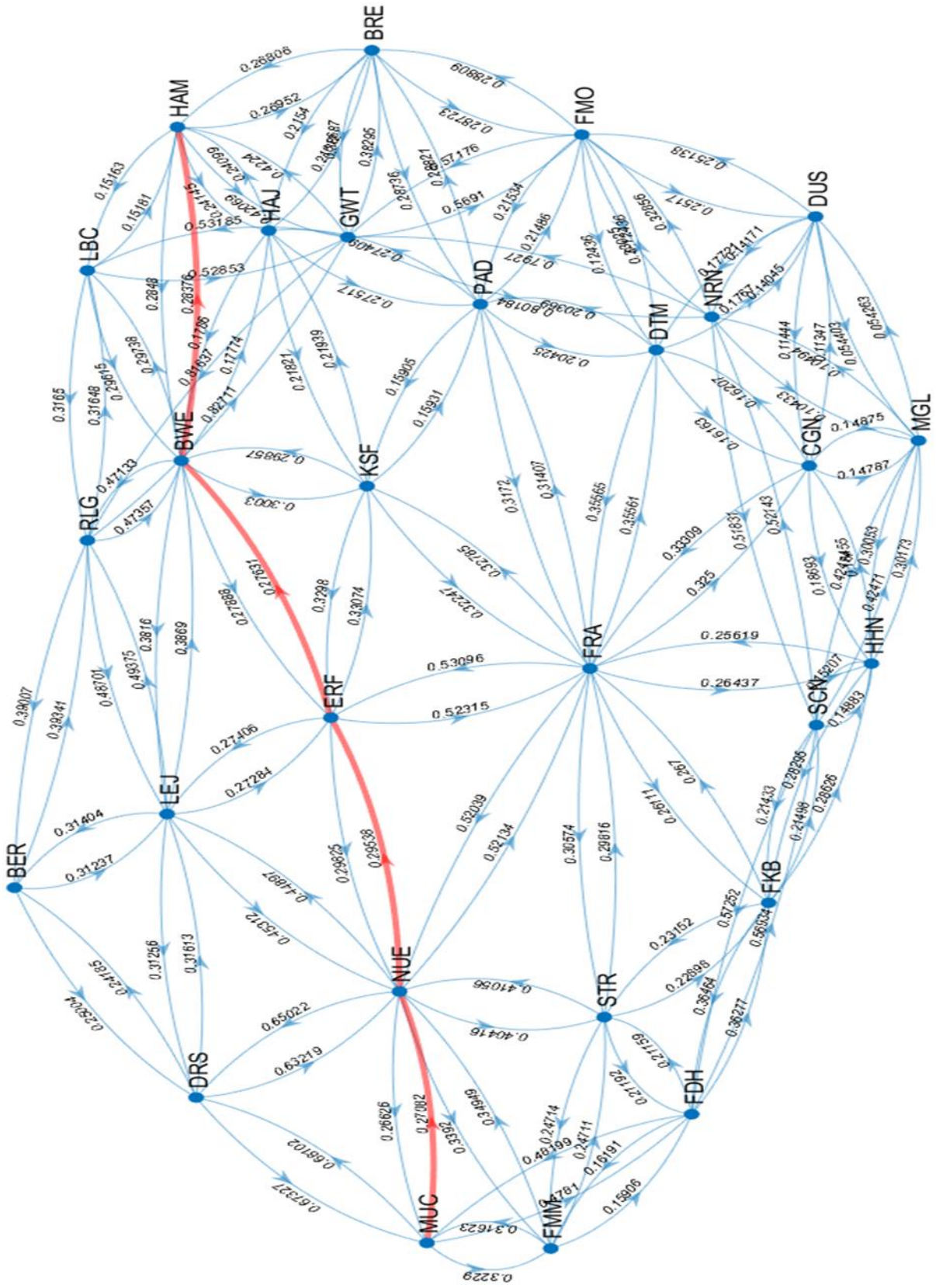
Jelikož geografické podmínky ve zkoumané oblasti nedávají vzniknout větru s dostatečně vysokou rychlostí, vliv větru na výslednou podobu cesty je minimální (resp. téměř žádný) – cesta zůstává pro oba problémy totožná (jak znázorňují Obrázek 16 a Obrázek 17).



Obrázek 16 – Cesta vygenerovaná Fordovým algoritmem pro graf nezohledňující vlivy větru [autor]

Ohodnocení	0	0.9262	1.2337	1.1274	1.1197	0.5696
Předchůdce	0	17	5	25	28	10

Tabulka 2– Vektor délek minimálních cest do jednotlivých vrcholů a vektor předchůdců pro graf nezohledňující vlivy větru [autor]



Obrázek 17 – Cesta vygenerovaná Fordovým algoritmem pro graf zohledňující vlivy větru [autor]

Ohodnocení	0	0.9229	1.2306	1.1263	1.1172	0.5700
Předchůdce	0	17	5	25	28	10

Tabulka 3 – Vektor délek minimálních cest do jednotlivých vrcholů a vektor předchůdců pro graf zohledňující vlivy větru [autor]

Jelikož prakticky získaná data ovlivnila chod algoritmu a výslednou cestu na zvoleném grafu jen minimálně, je do budoucna nutno zaměřit se více na konstrukci sítě/grafu zkoumaného prostředí. Využitý algoritmus by bylo vhodnější aplikovat na grafy s vyšší hustotou pokrytí vrcholy tak, aby se vlivy větru na hledání minimálních cest znatelně projevilo. Příkladem prostředí, které by mohlo být vhodným vstupem algoritmu je například Obrázek 3 – Mapa FRA České republiky pro letové hladiny FL245 – FL660 [zdroj: 19].

Další možnou modifikací by mohlo představovat využití či důraznější zaměření na zprávy typu SPECI (namísto zpráv typu METAR), které jsou vydávány pouze v případech mimořádných meteorologických jevů, naměřené vstupní hodnoty pro zpracování Fordovým (případně jakýmkoliv jiným) algoritmem by dosahovaly výrazně vyšších úrovní, což by se při volbě správně zkonstruovaného grafu mělo výrazněji projevit.

Závěr

Letecká doprava je jedním z klíčových pilířů moderního světového obchodu a cestování. Rychlost, pohodlí a dostupnost, které letectví poskytuje, jsou nezbytné pro propojení odlehlých destinací a podporu mezinárodního obchodu a turismu. Nicméně, efektivní provoz letového provozu vyžaduje nejen dokonalou koordinaci mezi leteckými společnostmi, letišti a řídícími věžemi, ale také optimalizaci letových tras.

Optimalizace letových tras je zásadní pro minimalizaci nákladů, zkrácení doby letu, snížení spotřeby paliva a zvýšení kapacity letecké dopravy. Důsledně plánované trasy přispívají k efektivitě leteckého provozu, minimalizaci zpoždění a zlepšení spokojenosti cestujících. Optimalizace letových tras je také klíčovým faktorem pro řízení logistiky a přepravy nákladů, přičemž přispívá k urychlení přepravy zboží a zajištění bezproblémového průběhu dodavatelského řetězce.

Plánování letových tras je ovšem složitý problém, který vyžaduje analýzu různých faktorů, jako jsou geografické podmínky, provozní omezení, meteorologické podmínky, dostupnost letišť a potřeby cestujících. Řešení tohoto problému využívá metodologie teorie grafů, která poskytuje vhodný matematický rámec pro modelování leteckých sítí a hledání optimálních tras.

Cílem této práce byla analýza možností využití Fordova algoritmu pro plánování letových tras a vytvoření modifikací algoritmu a jeho vstupních parametrů pro aplikaci na tuto problematiku v rámci koncepce Free Route Airspace.

Pro řešení problematiky plánování letových tras je v současné době využívána řada metod z různých vědních oborů a disciplín, nejen Teorie grafů. Mezi tyto metody patří základní algoritmy pro hledání minimálních cest v grafu jako jsou například Fordův, Floydův či Dijkstrův algoritmus, jejich modifikace doplněné o heuristické procesy (A* algoritmus) či řádově složitější způsoby výpočtu jako jsou například Algoritmy pro potenciální pole či genetické algoritmy.

Samotný Fordův algoritmus lze aplikovat zejména na grafy s nízkou složitostí a případně modifikovat vstupní hodnoty tak, aby algoritmus zohledňoval další faktory přítomné v průběhu pohybu letadla, jako jsou například efekty rychlosti a směru větru, dohlednost či teplota. Jednoduchost algoritmu dále umožňuje jeho velmi snadnou implementaci v řadě programovacích jazyků, například v prostředí MATLAB, ve kterém byl algoritmus aplikován pro potřeby bakalářské práce.

Praktická implementace algoritmu využívá možnosti stažení veškerých relevantních dat – zde souřadnice orientačních bodů a korespondující METAR zprávy, které byli jednotlivým bodům přiřazeny za využití kódů ICAO. Aplikací Voroného diagramů a Delaného triangulace lze sestrojit demonstrativní neorientovaný graf a využitím vzorce pro poloviční versus sinus jej doplnit o distanční ohodnocení jednotlivých hran, čímž je vytvořen vstupní parametr Fordova algoritmu. Algoritmus lze nadále pomocí dat získaných ze zpráv METAR modifikovat o vlivy vybraných meteorologických jevů – rychlost a směr větru. Aplikací vzorce pro modifikovanou dobu letu je demonstrativní vstupní graf transformován na digraf, což umožňuje zisk přesnějších výsledků a dosažení vyšších časových úspor.

Na základě výstupů získaných aplikací algoritmu na demonstrativní příklad lze pozorovat vznik časových úspor při modifikaci algoritmu o vlivy větru, na vybraném příkladě jsou tyto úspory ovšem minimální. Pro efektivní využití těchto modifikací je tudíž klíčová správná konstrukce vstupního grafu, kterou se tato práce do hloubky nezabývá, jelikož vlivy větru na minimální cestu a dobu trvání letu (popřípadě další vybrané parametry) jsou znatelné pouze ve správném měřítku. Budoucí výzkum by měl tedy být zaměřen více na volbu či tvorbu vstupního grafu, kde příkladem takto vhodného vstupního grafu může být zejména mapa vybraného FRA či trasa transatlantického letu, kde se vlivy větru mohou znatelně projevit.

V případě zohlednění nedostatků vstupních parametrů algoritmu v dalším výzkumu by mohlo dojít k jeho aplikaci na praktické problémy a dosažení významných úspor doby letu a spotřeby paliva, a to v oblasti nákladní i osobní letecké dopravy zejména na dlouhé vzdálenosti.

Seznam použité literatury a jiných zdrojů

[1] DE, Luca a Giorgio GUGLIERI. Advanced Graph Search Algorithms for Path Planning of Flight Vehicles. In: AGARWAL, Ramesh, ed. *Recent Advances in Aircraft Technology* [online]. InTech, 2012, 2012-02-24 [cit. 2023-10-06]. ISBN 978-953-51-0150-5. Dostupné z: doi:10.5772/370336

[2] Jensen, Casper Kehlet; Chiarandini, Marco; Larsen, Kim S. Flight Planning in Free Route Airspaces. In: 17th Workshop on Algorithmic Approaches for Transportation Modelling, Optimization, and Systems (ATMOS 2017). Dagstuhl, 2017, 2017-09-04 [cit. 2023-10-06] ISBN 978-3-95977-042-2. Dostupné z: doi: 10.4230/OASIcs.ATMOS.2017.14

[3] De Filippis, L., Guglieri, G., Quagliotti, F. (2009). Flight Analysis and Design for Mini-UAVs.Proceedings of XX AIDAA Congress, Milano, Italy.

[4] De Filippis, L., Guglieri, G., Quagliotti, F. (2010). A minimum risk approach for path planning of UAVs. Journal of Intelligent and Robotic Systems, Springer,

[5] De Filippis, L., Guglieri, G. & Quagliotti, F. (2011). Path Planning strategies for UAVs in 3D environments. Journal of Intelligent and Robotic Systems, Springer.

[6] Jun, M. & D'Andrea, R. (2002). Path Planning for Unmanned Aerial Vehicles in Uncertain and Adversarial Environments. Models, Applications and Algorithms, Kluwer Academic Press.

[7] Pfeiffer, B., Batta, R., Klamroth, K. & Nagi, R. (2008). Path Planning for UAVs in the Presence of Threat Zones Using Probabilistic Modelling. In: Handbook of Military Industrial Engineering, Taylor and Francis, USA.

[8] Horner, D., P. & Healey, A., J. (2004). Use of artificial potential fields for UAV guidance and optimization of WLAN communications. Autonomous Underwater Vehicles, 2004
IEEE/OES, vol., no.

[9] Ma, X. & Castanon, D.A. (2006). Receding Horizon Planning for Dubins Traveling Salesman Problems. Proceedings of IEEE Conference on Decision and Control, San Diego, USA.

[10] Chen, Qingyang & Lu, Ya-fei & Jia, Gao-wei & Li, Yue & Zhu, Bing-jie & Lin, Jun-can. (2018). Path planning for UAVs formation reconfiguration based on Dubins trajectory. Journal of Central South University. 25. 2664-2676. 10.1007/s11771-018-3944-z.

[11] Mujumdar, Anusha & Padhi, R. (2009). Nonlinear Geometric and Differential Geometric Guidance of UAVs for Reactive Collision Avoidance. Journal of Guidance, Control, and Dynamics. 34. 69. 10.2514/1.50923.

[12] Salamat, Sahand & Khaleghi, Behnam & Imani, Mohsen & Rosing, Tajana. (2019). Workload-Aware Opportunistic Energy Efficiency in Multi-FPGA Platforms.

[13] Free Route Airspace – Giving Users the Freedom to plan a route in Europe's airspace. EUROCONTROL [online]. Brusel, 2018 [cit. 2023-10-06] Dostupné z: https://www.eurocontrol.int/

[14] About ICAO. International Civil Aviation Organization [online]. Montreal [cit. 2023-10-06]. Dostupné z: https://www.icao.int

[15] *Eurocontrol Free Route Airspace (FRA)* [online]. EUROCONTROL. [cit. 2023-10-06]. Dostupné z: https://www.eurocontrol.int/concept/Free Route-airspace

[16] Delta Oscar Uniform (1966a, April 1). *What methodology did ICAO use to create the overall code nomenclature for airports?* Aviation Stack Exchange. [online]. Aviation Stack Exchange. [cit. 2023-10-06]. Dostupné z: https://aviation.stackexchange.com/questions/68163/what-methodology-did-icao-use-to-create-the-overall-code-nomenclature-for-airpor

[17] Meteo LKPR - 21.06.2023 12:00:39. [online]. Řízení letového provozu. [cit. 2023-10-06]. Dostupné z: https://meteo.rlp.cz/LKPR_meteo.htm

[18] Airports in Germany | gifex. [online]. Gifex. [cit. 2023-10-06]. Dostupné z: https://www.gifex.com/detail2-en/2011-05-27-13798/Airports_in_Germany.html

[19] *Z českého nebe a leteckých map zmizely letové tratě*. [online] Ministerstvo dopravy ČR – Média a tiskové zprávy. [cit. 2023-10-06]. Dostupné z: https://www.mdcr.cz/Media/Media-a-tiskove-zpravy/Z-ceskeho-nebe-a-leteckych-map-zmizely-letove-trat

[20] *Bellman Ford's algorithm*. Programiz. [online]. Programiz. [cit. 2023-10-06]. Dostupné z: https://www.programiz.com/dsa/bellman-ford-algorithm

[21] *Delaunay triangulation*. from Wolfram MathWorld. [online]. Wolfram alpha. [cit. 2023-10-06]. Dostupné z: https://mathworld.wolfram.com/DelaunayTriangulation.html

[22] *Lex – 32008R1008 – en – EUR-lex*. EUR. [online]. Úřední věštník Evropské Unie. [cit. 2023-10-06]. Dostupné z: https://eur-lex.europa.eu/legal-content/cs/TXT/?uri=CELEX%3A32016R0679

[23] *Lex – 32004R0549 – en – EUR-lex*. EUR. [online]. Úřední věštník Evropské Unie. [cit. 2023-10-06]. Dostupné z: https://eur-lex.europa.eu/legal-content/CS/ALL/?uri=CELEX%3A32004R0549

[24] Aurenhammer, F., Klein, R. and Lee, D.T., 2013. Voronoi Diagrams and Delaunay Triangulations. World Scientific Publishing. ISBN 978-9814447638

[25] Strößner, Corina. (2022). Criteria for naturalness in conceptual spaces. Synthese. 200. 10.1007/s11229-022-03610-4.

[26] Korn, Grandino Arthur; Korn, Theresa M. (2000) [1922]. "Appendix B: B9. Plane and Spherical Trigonometry: Formulas Expressed in Terms of the Haversine Function". Mathematical handbook for scientists and engineers: Definitions, theorems, and formulas for reference and review (3rd ed.). Mineola, New York: Dover Publications. pp. 892–893. ISBN 978-0-486-41147-7.

[27] Chan, Timothy M. (January 2010), "More algorithms for all-pairs shortest paths in weighted graphs", SIAM Journal on Computing, 39 (5): 2075–2089, CiteSeerX 10.1.1.153.6864, doi:10.1137/08071990x.

[28] Bellman, Richard (1958). "On a routing problem". Quarterly of Applied Mathematics. 16: 87–90. doi:10.1090/qam/102435. MR 0102435.

Obrazová příloha

Seznam Tabulek

Seznam vzorců